Springer Handbook of Speech Processing

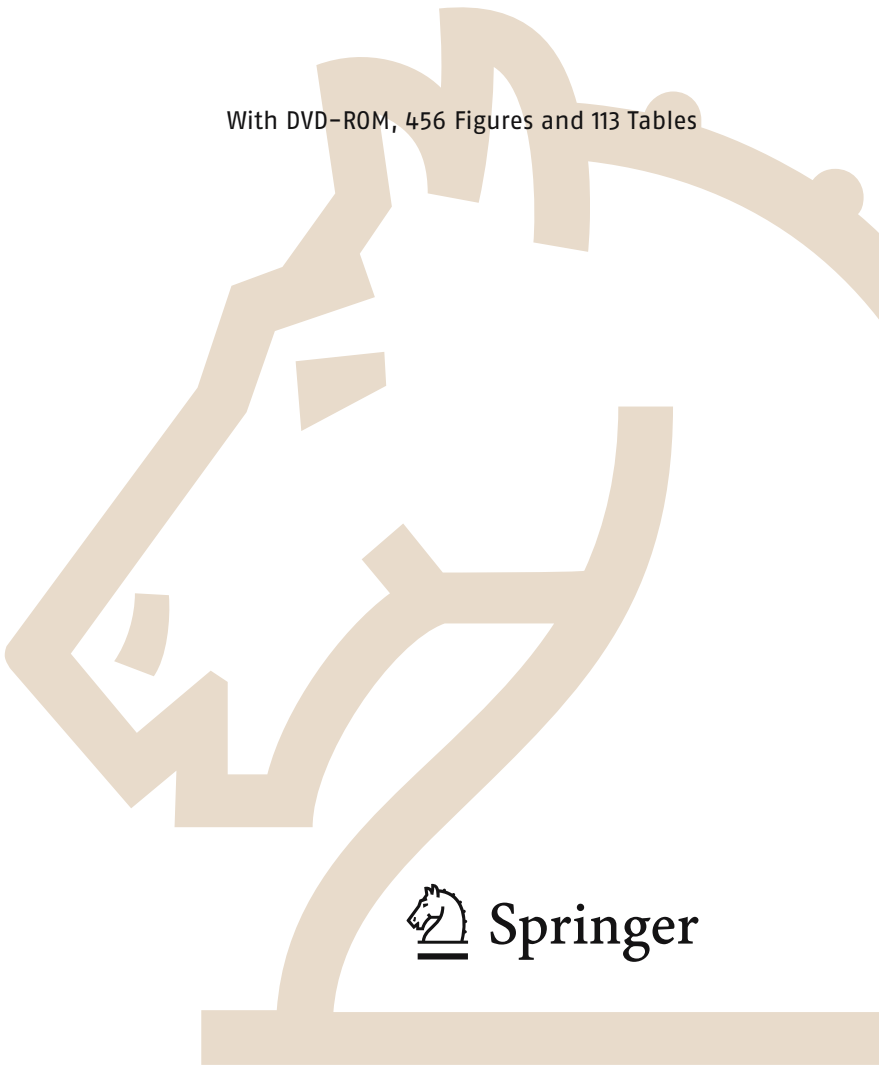
Springer Handbooks provide a concise compilation of approved key information on methods of research, general principles, and functional relationships in physical sciences and engineering. The world's leading experts in the fields of physics and engineering will be assigned by one or several renowned editors to write the chapters comprising each volume. The content is selected by these experts from Springer sources (books, journals, online content) and other systematic and approved recent publications of physical and technical information.

The volumes are designed to be useful as readable desk reference books to give a fast and comprehensive overview and easy retrieval of essential reliable key information, including tables, graphs, and bibliographies. References to extensive sources are provided.

Springer Handbook of Speech Processing

Jacob Benesty, M. Mohan Sondhi, Yiteng Huang (Eds.)

With DVD-ROM, 456 Figures and 113 Tables



Springer

Editors:

Jacob Benesty
INRS-EMT, University of Quebec
800 de la Gauchetiere Ouest, Suite 6900
Montreal, Quebec, H5A 1K6, Canada
benesty@emt.inrs.ca

M. Mohan Sondhi
Avayalabs Research
233 Mount Airy Road
Basking Ridge, NJ 07920, USA
mms@research.avayalabs.com

Yiteng Huang
Bell Laboratories, Alcatel-Lucent
600 Mountain Avenue
Murray Hill, NJ 07974, USA
arden_huang@ieee.org

Library of Congress Control Number: 2007931999

ISBN: 978-3-540-49125-5 e-ISBN: 978-3-540-49127-9

Springer is a part of Springer Science+Business Media

springer.com

Typesetting and production:
LE-TEX Jelonek, Schmidt&Vöckler GbR, Leipzig
Senior Manager Springer Handbook: Dr. W. Skolaut, Heidelberg
Typography and layout: schreiberVIS, Seeheim
Illustrations: Hippmann GbR, Schwarzenbruck
Cover design: eStudio Calamar Steinen, Barcelona
Cover production: WMXDesign GmbH, Heidelberg
Printing and binding: Stürtz GmbH, Würzburg

Printed on acid free paper

SPIN 11544036 60/3180/YL 5 4 3 2 1 0

Foreword

J. L. Flanagan
Professor Emeritus
Electrical and Computer Engineering
Rutgers University

Over the past three decades digital signal processing has emerged as a recognized discipline. Much of the impetus for this advance stems from research in representation, coding, transmission, storage and reproduction of speech and image information. In particular, interest in voice communication has stimulated central contributions to digital filtering and discrete-time spectral transforms.

This dynamic development was built upon the convergence of three then-evolving technologies: (i) sampled-data theory and representation of information signals (which led directly to digital telecommunication that provides signal quality independent of transmission distance); (ii) electronic binary computation (aided in early implementation by pulse-circuit techniques from radar design); and, (iii) invention of solid-state devices for exquisite control of electronic current (transistors – which now, through microelectronic materials, scale to systems of enormous size and complexity). This timely convergence was soon followed by optical fiber methods for broadband information transport.

These advances impact an important aspect of human activity – information exchange. And, over man's existence, speech has played a principal role in human communication. Now, speech is playing an increasing role in human interaction with complex information systems. Automatic services of great variety exploit the comfort of voice exchange, and, in the corporate sector, sophisticated audio/video teleconferencing is reducing the necessity of expensive, time-consuming business travel. In each instance an overarching target is a user environment that captures some of the naturalness and spatial realism of face-to-face communication. Again, speech is a core element, and new understanding from diverse research sectors can be brought to bear.

Editors-in-Chief Benesty, Sondhi and Huang have organized a timely engineering handbook to answer this need. They have assembled a remarkable compendium of current knowledge in speech processing. And, this accumulated understanding can be focused upon enlarging the human capacity to deal with a world ever increasing in complexity. Benesty, Sondhi and Huang are renowned researchers in their own right, and they have attracted an international cadre of over 80 fellow authors and collaborators who constitute a veritable *Who's Who* of world leaders in speech processing research. The resulting book provides under one cover authoritative treatments that commence with the basic physics and psychophysics of speech and hearing, and range through the related topics of computational tools, coding, synthesis, recognition, and signal enhancement, concluding with discussions on capture and projection of sound in enclosures. The book can be expected to become a valuable resource for researchers, engineers and speech scientists throughout the global community. It should equally serve teachers and students in human communication, especially delimiting knowledge frontiers where graduate thesis research may be appropriate.

Warren, New Jersey
October 2007

Jim Flanagan

Preface

The achievement of this Springer Handbook is the result of a wonderful journey that started in March 2005 at the 30th International Conference on Acoustics, Speech, and Signal Processing (ICASSP). Two of the editors-in-chief (Benesty and Huang) met in one of the long corridors of the Pennsylvania Convention Center in Philadelphia with Dr Dieter Merkle from Springer. Together we had a very nice discussion about the conference and immediately an idea came up for a handbook. After a short discussion we converged without too much hesitation on a handbook of *speech processing*. It was quite surprising to see that, even after 30 years of ICASSP and more than half a century of research in this fundamental area, there was still no major book summarizing the important aspects of speech processing. We thought that the time was ripe for such a large project. Soon after we got home, a third editor-in-chief (Sondhi) joined the efforts.

Jacob Benesty

M. Mohan Sondhi

Yiteng Huang

We had a very clear objective in our minds: to summarize, in a reasonable number of pages, the most important and useful aspects of speech processing. The content was then organized accordingly. This task was not easy since we had to find a good balance between feasible ideas and new trends. As we all know, practical ideas can be viewed as *old stuff* while emerging ideas can be criticized for not having passed the test of time; we hope that we have succeeded in finding a good compromise. For this we relied on many authors who are well established and are recognized as experts in their field, from all over the world, and from academia as well as from industry.

From *simple* consumer products such as cell phones and MP3 players to more-sophisticated projects such as human–machine interfaces and robots that can obey orders, speech technologies are now everywhere. We believe that it is just a matter of time before more applications of the science of speech become impossible to miss in our daily life. So we believe that this Springer Handbook will play a fundamental role in the sustainable progress of speech research and development.

This handbook is targeted at three categories of readers: graduate students of speech processing, professors and researchers in academia and research labs who are active in this field, and engineers in industry who need to understand or implement specific algorithms for their speech-related products. The handbook could also be used as a text for one or more graduate courses on signal processing for speech and various aspects of speech processing and applications.

For the completion of such an ambitious project we have many people to thank. First, we would like to thank the many authors who did a terrific job in delivering very high-quality chapters. Second, we are very grateful to the members of the editorial board who helped us so much in organizing the content and structure of this book, taking part in all phases of this project from conception to completion. Third, we would like to thank all the reviewers, who helped us to improve the quality of the material. Last, but not least, we would like to thank the Springer team for their availability and very professional work. In particular, we appreciated the help of Dieter Merkle, Christoph Baumann, Werner Skolaut, Petra Jantzen, and Claudia Rau.

We hope this Springer Handbook will inspire many great minds to find new research ideas or to implement algorithms in products.

Montreal, Basking Ridge, Murray Hill
October 2007

Jacob Benesty
M. Mohan Sondhi
Yiteng Huang

List of Editors

Editors-in-Chief

Jacob Benesty, Montreal
M. Mohan Sondhi, Basking Ridge
Yiteng (Arden) Huang, Murray Hill

Part Editors

Part A: Production, Perception, and Modeling of Speech

M. M. Sondhi, Basking Ridge

Part B: Signal Processing for Speech

Y. Huang, Murray Hill; J. Benesty, Montreal

Part C: Speech Coding

W. B. Kleijn, Stockholm

Part D: Text-to-Speech Synthesis

S. Narayanan, Los Angeles

Part E: Speech Recognition

L. Rabiner, Piscataway; B.-H. Juang, Atlanta

Part F: Speaker Recognition

S. Parthasarathy, Sunnyvale

Part G: Language Recognition

C.-H. Lee, Atlanta

Part H: Speech Enhancement

J. Chen, Murray Hill; S. Gannot, Ramat-Gan; J. Benesty, Montreal

Part I: Multichannel Speech Processing

J. Benesty, Montreal; I. Cohen, Haifa; Y. Huang, Murray Hill

List of Authors

Alex Acero
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA
e-mail: *alexac@microsoft.com*

Jont B. Allen
University of Illinois
ECE
Urbana, IL 61801, USA
e-mail: *JontAllen@ieee.org*

Jacob Benesty
University of Quebec
INRS-EMT
800 de la Gauchetiere Ouest
Montreal, Quebec H5A 1K6, Canada
e-mail: *benesty@emt.inrs.ca*

Frédéric Bimbot
IRISA (CNRS & INRIA) - METISS
Pièce C 320 - Campus Universitaire de Beaulieu
35042 Rennes, France
e-mail: *bimbot@irisa.fr*

Thomas Brand
Carl von Ossietzky Universität Oldenburg
Sektion Medizinphysik
Haus des Hörens, Marie-Curie-Str. 2
26121 Oldenburg, Germany
e-mail: *thomas.brand@uni-oldenburg.de*

Nick Campbell
Knowledge Creating Communication Research Centre
Acoustics & Speech Research Project, Spoken Language Communication Group
2-2-2 Hikaridai
619-0288 Keihanna Science City, Japan
e-mail: *nick@nict.go.jp*

William M. Campbell
MIT Lincoln Laboratory
Information Systems Technology Group
244 Wood Street
Lexington, MA 02420-9108, USA
e-mail: *wcampbell@ll.mit.edu*

Rolf Carlson
Royal Institute of Technology (KTH)
Department of Speech, Music and Hearing
Lindstedtsvägen 24
10044 Stockholm, Sweden
e-mail: *rolf@speech.kth.se*

Jingdong Chen
Bell Laboratories
Alcatel-Lucent
600 Mountain Ave
Murray Hill, NJ 07974, USA
e-mail: *jingdong@research.bell-labs.com*

Juin-Hwey Chen
Broadcom Corp.
5300 California Avenue
Irvine, CA 92617, USA
e-mail: *rchen@broadcom.com*

Israel Cohen
Technion–Israel Institute of Technology
Department of Electrical Engineering
Technion City
Haifa 32000, Israel
e-mail: *icohen@ee.technion.ac.il*

Jordan Cohen
SRI International
300 Ravenswood Drive
Menlo Park, CA 94019, USA
e-mail: *jrc@speech.sri.com*

Corinna Cortes
Google, Inc.
Google Research
76 9th Avenue, 4th Floor
New York, NY 10011, USA
e-mail: *corinna@google.com*

Eric J. Diethorn
Avaya Labs Research
Multimedia Technologies Research Department
233 Mt. Airy Road
Basking Ridge, NJ 07920, USA
e-mail: *ejd@avaya.com*

Simon Doclo
Katholieke Universiteit Leuven
Department of Electrical Engineering (ESAT-SCD)
Kasteelpark Arenberg 10 bus 2446
3001 Leuven, Belgium
e-mail: *simon.doclo@esat.kuleuven.be*

Jasha Droppo
Microsoft Research
Speech Technology Group
One Microsoft Way
Redmond, WA 98052, USA
e-mail: *jdroppo@microsoft.com*

Thierry Dutoit
Faculté Polytechnique de Mons FPMs
TCTS Laboratory
Bvd Dolez, 31
7000 Mons, Belgium
e-mail: *thierry.dutoit@fpms.ac.be*

Gary W. Elko
mh acoustics LLC
25A Summit Ave
Summit, NJ 07901, USA
e-mail: *gwe@mhacoustics.com*

Sadaoki Furui
Tokyo Institute of Technology Street
Department of Computer Science
2-12-1 Ookayama, Meguro-ku
152-8552 Tokyo, Japan
e-mail: *furui@cs.titech.ac.jp*

Sharon Gannot
Bar-Ilan University
School of Electrical Engineering
Ramat-Gan 52900, Israel
e-mail: *gannot@eng.biu.ac.il*

Mazin E. Gilbert
AT&T Labs, Inc., Research
180 Park Ave.
Florham Park, NJ 07932, USA
e-mail: *mazin@research.att.com*

Michael M. Goodwin
Creative Advanced Technology Center
Audio Research
1500 Green Hills Road
Scotts Valley, CA 95066, USA
e-mail: *mgoodwin@atc.creative.com*

Volodya Grancharov
Multimedia Technologies
Ericsson Research, Ericsson AB
Torshamnsgatan 23, Kista, KI/EAB/TVA/A
16480 Stockholm, Sweden
e-mail: *volodya.grancharov@ericsson.com*

Björn Granström
Royal Institute of Technology (KTH)
Department for Speech, Music and Hearing
Lindstedsvägen 24
10044 Stockholm, Sweden
e-mail: *bjorn@speech.kth.se*

Patrick Haffner
AT&T Labs-Research
IP and Voice Services
200 S Laurel Ave.
Middletown, NJ 07748, USA
e-mail: *haffner@research.att.com*

Roar Hagen
Global IP Solutions
Magnus Ladulsgatan 63B
118 27 Stockholm, Sweden
e-mail: *roar.hagen@gipscorp.com*

Mary P. Harper
University of Maryland
Center for Advanced Study of Language
7005 52nd Avenue
College Park, MD 20742, USA
e-mail: *mharper@casl.umd.edu*

Jürgen Herre
Fraunhofer Institute for Integrated Circuits
(Fraunhofer IIS)
Audio and Multimedia
Am Wolfsmantel 33
91058 Erlangen, Germany
e-mail: *hrr@iis.fraunhofer.de*

Wolfgang J. Hess
University of Bonn
Institute for Communication Sciences, Dept. of Communication, Language, and Speech
Poppelsdorfer Allee 47
53115 Bonn, Germany
e-mail: *wgh@ifk.uni-bonn.de*

Kiyoshi Honda
Université de la Sorbonne Nouvelle-Paris III
Laboratoire de Phonétique et de Phonologie, ATR Cognitive Information Laboratories
UMR-7018-CNRS, 46, rue Barrault
75634 Paris, France
e-mail: *honda@atr.jp*

Yiteng (Arden) Huang
Bell Laboratories
Alcatel-Lucent
600 Mountain Avenue
Murray Hill, NJ 07974, USA
e-mail: *arden_huang@ieee.org*

Matthieu Hébert
Network ASR Core Technology
Nuance Communications
1500 Université
Montréal, Québec H3A-3S7, Canada
e-mail: *hebert@nuance.com*

Biing-Hwang Juang
Georgia Institute of Technology
School of Electrical & Computer Engineering
777 Atlantic Dr. NW
Atlanta, GA 30332-0250, USA
e-mail: *juang@ece.gatech.edu*

Tatsuya Kawahara
Kyoto University
Academic Center for Computing and Media Studies
Sakyo-ku
606-8501 Kyoto, Japan
e-mail: *kawahara@i.kyoto-u.ac.jp*

Ulrik Kjems
Oticon A/S
9 Kongebakken
2765 Smørum, Denmark
e-mail: *uk@oticon.dk*

Esther Klabbers
Oregon Health & Science University
Center for Spoken Language Understanding, OGI School of Science and Engineering
20000 NW Walker Rd
Beaverton, OR 97006, USA
e-mail: *klabbers@cslu.ogi.edu*

W. Bastiaan Kleijn
Royal Institute of Technology (KTH)
School of Electrical Engineering, Sound and Image Processing Lab
Osquldas väg 10
10044 Stockholm, Sweden
e-mail: *bastiaan.kleijn@ee.kth.se*

Birger Kollmeier
Universität Oldenburg
Medizinische Physik
26111 Oldenburg, Germany
e-mail: *birger.kollmeier@uni-oldenburg.de*

Ermin Kozica
Royal Institute of Technology (KTH)
School of Electrical Engineering, Sound and Image Processing Laboratory
Osquldas väg 10
10044 Stockholm, Sweden
e-mail: *ermin.kozica@ee.kth.se*

Sen M. Kuo
Northern Illinois University
Department of Electrical Engineering
DeKalb, IL 60115, USA
e-mail: *kuo@ceet.niu.edu*

Jan Larsen
Technical University of Denmark
Informatics and Mathematical Modelling
Richard Petersens Plads
2800 Kongens Lyngby, Denmark
e-mail: *jl@imm.dtu.dk*

Chin-Hui Lee
Georgia Institute of Technology
School of Electrical and Computer Engineering
777 Atlantic Drive NW
Atlanta, GA 30332-0250, USA
e-mail: *chl@ece.gatech.edu*

Haizhou Li
Institute for Infocomm Research
Department of Human Language Technology
21 Heng Mui Keng Terrace
Singapore, 119613
e-mail: *hli@i2r.a-star.edu.sg*

Jan Linden
Global IP Solutions
301 Brannan Street
San Francisco, CA 94107, USA
e-mail: *jan.linden@gipscorp.com*

Manfred Lutzky
Fraunhofer Integrated Circuits (IIS)
Multimedia Realtime Systems
Am Wolfsmantel 33
91058 Erlangen, Germany
e-mail: *manfred.lutzky@iis.fraunhofer.de*

Bin Ma
Human Language Technology
Institute for Infocomm Research
21 Heng Mui Keng Terrace
Singapore, 119613
e-mail: *mabin@i2r.a-star.edu.sg*

Michael Maxwell
University of Maryland
Center for Advanced Study of Language
Box 25
College Park, MD 20742, USA
e-mail: *mmaxwell@casl.umd.edu*

Alan V. McCree
MIT Lincoln Laboratory
Department of Information Systems Technology
244 Wood Street
Lexington, MA 02420-9185, USA
e-mail: *mccree@ll.mit.edu*

Bernd Meyer
Carl von Ossietzky Universität Oldenburg
Medical Physics Section, Haus des Hörens
Marie-Curie-Str. 2
26121 Oldenburg, Germany
e-mail: *bernd.meyer@uni-oldenburg.de*

Jens Meyer
mh acoustcis
25A Summit Ave.
Summit, NJ 07901, USA
e-mail: *jmm@mhacoustics.com*

Taniya Mishra
Oregon Health and Science University
Center for Spoken Language Understanding,
Computer Science and Electrical Engineering, OGI
School of Science and Engineering
20000 NW Walker Road
Beaverton, OR 97006, USA
e-mail: *mishra@cslu.ogi.edu*

Mehryar Mohri
Courant Institute of Mathematical Sciences
251 Mercer Street
New York, NY 10012, USA
e-mail: *mohri@cs.nyu.edu*

Marc Moonen
Katholieke Universiteit Leuven
Electrical Engineering Department ESAT/SISTA
Arenberg 10
3001 Leuven, Belgium
e-mail: *marc.moonen@esat.kuleuven.be*

Dennis R. Morgan
Bell Laboratories, Alcatel-Lucent
700 Mountain Avenue 2D-537
Murray Hill, NJ 07974-0636, USA
e-mail: *drrm@bell-labs.com*

David Nahamoo
IBM Thomas J. Watson Research Center
PO BOX 218
Yorktown Heights, NY 10598, USA
e-mail: *nahamoo@us.ibm.com*

Douglas O'Shaughnessy
Université du Québec
INRS Énergie, Matériaux et Télécommunications
(INRS-EMT)
800, de la Gauchetiere Ouest
Montréal, Québec H5A 1K6, Canada
e-mail: *dougo@emt.inrs.ca*

Lucas C. Parra
Steinman Hall, The City College of New York
Department of Biomedical Engineering
140th and Convent Ave
New York, NY 10031, USA
e-mail: *parra@ccny.cuny.edu*

Sarangarajan Parthasarathy
Yahoo!, Applied Research
1MC 743, 701 First Avenue
Sunnyvale, CA 94089-0703, USA
e-mail: *parthas@yahoo-inc.com*

Michael Syskind Pedersen
Oticon A/S
Kongebakken 9
2765 Smørum, Denmark
e-mail: *msp@oticon.dk*

Fernando Pereira
University of Pennsylvania
Department of Computer and Information Science
305 Levine Hall, 3330 Walnut Street
Philadelphia, PA 19104, USA
e-mail: *pereira@cis.upenn.edu*

Michael Picheny
IBM Thomas J. Watson Research Center
Yorktown Heights, NY 10598, USA
e-mail: *Picheny@us.ibm.com*

Rudolf Rabenstein
University Erlangen-Nuremberg
Electrical Engineering, Electronics, and
Information Technology
Cauerstrasse 7/LMS
91058 Erlangen, Germany
e-mail: *rabe@LNT.de*

Lawrence Rabiner
Rutgers University
Department of Electrical and Computer
Engineering
96 Frelinghuysen Road
Piscataway, NJ 08854, USA
e-mail: *lrr@caip.rutgers.edu*

Douglas A. Reynolds
Massachusetts Institute of Technology
Lincoln Laboratory, Information Systems
Technology Group
244 Wood Street
Lexington, MA 02420-9108, USA
e-mail: *dar@ll.mit.edu*

Michael Riley
Google, Inc., Research
111 Eighth AV
New York, NY 10011, USA
e-mail: *riley@google.com*

Aaron E. Rosenberg
Rutgers University
Center for Advanced Information Processing
96 Frelinghuysen Road
Piscataway, NJ 08854-8088, USA
e-mail: *aer@caip.rutgers.edu*

Salim Roukos
IBM T. J. Watson Research Center
Multilingual NLP Technologies
Yorktown Heights, NY 10598, USA
e-mail: *roukos@us.ibm.com*

Jan van Santen
Oregon Health And Science University
OGI School of Science and Engineering,
Department of Computer Science and Electrical
Engineering
20000 NW Walker Rd
Beaverton, OR 97006-8921, USA
e-mail: *vansanten@cslu.ogi.edu*

Ronald W. Schafer
Hewlett-Packard Laboratories
1501 Page Mill Road
Palo Alto, CA 94304, USA
e-mail: *ron.schafer@hp.com*

Juergen Schroeter
AT&T Labs - Research
Department of Speech Algorithms and Engines
180 Park Ave
Florham Park, NJ 07932, USA
e-mail: *schroeter@att.com*

Stephanie Seneff
Massachusetts Institute of Technology
Computer Science and Artificial
Intelligence Laboratory
32 Vassar Street
Cambridge, MA 02139, USA
e-mail: *senef@csail.mit.edu*

Wade Shen
Massachusetts Institute of Technology
Communication Systems, Information Systems
Technology, Lincoln Laboratory
244 Wood Street
Lexington, MA 02420-9108, USA
e-mail: *swade@ll.mit.edu*

Elliot Singer
Massachusetts Institute of Technology
Information Systems Technology Group, Lincoln
Laboratory
244 Wood Street
Lexington, MA 02420-9108, USA
e-mail: *es@ll.mit.edu*

Jan Skoglund
Global IP Solutions
301 Brannan Street
San Francisco, CA 94107, USA
e-mail: *jan.skoglund@gipscorp.com*

M. Mohan Sondhi
Avayalabs Research
233 Mount Airy Road
Basking Ridge, NJ 07920, USA
e-mail: *mms@research.avayalabs.com*

Sascha Spors
Deutsche Telekom AG, Laboratories
Ernst-Reuter-Platz 7
10587 Berlin, Germany
e-mail: *Sascha.Spors@telekom.de*

Ann Spriet
ESAT-SCD/SISTA, K.U. Leuven
Department of Electrical Engineering
Kasteelpark Arenberg 10
3001 Leuven, Belgium
e-mail: *ann.spriet@esat.kuleuven.be*

Richard Sproat
University of Illinois at Urbana-Champaign
Department of Linguistics
Urbana, IL 61801, USA
e-mail: *rws@uiuc.edu*

Yannis Stylianou
Institute of Computer Science
Heraklion, Crete 700 13, Greece
e-mail: *yannis@csd.uoc.gr*

Jes Thyssen
Broadcom Corporation
5300 California Avenue
Irvine, CA 92617, USA
e-mail: *jthyssen@broadcom.com*

Jay Wilpon
Research AT&T Labs
Voice and IP Services
Florham Park, NJ 07932, USA
e-mail: *jgw@research.att.com*

Jan Wouters
ExpORL, Department of Neurosciences, K.U. Leuven
O.& N2, Herestraat 49
3000 Leuven, Belgium
e-mail: *jan.wouters@med.kuleuven.be*

Arie Yeredor
Tel-Aviv University
Electrical Engineering - Systems
Tel-Aviv 69978, Israel
e-mail: *arie@eng.tau.ac.il*

Steve Young
Cambridge University Engineering Dept
Cambridge, CB21PZ, UK
e-mail: *sjy@eng.cam.ac.uk*

Victor Zue
Massachusetts Institute of Technology
CSAI Laboratory
32 Vassar Street
Cambridge, MA 02139, USA
e-mail: *zue@csail.mit.edu*

Contents

Part B Signal Processing for Speech

Part G Language Recognition

Part H Speech Enhancement

List of Abbreviations

2TS	two-tone suppression

A

ACELP	algebraic code excited linear prediction
ACF	autocorrelation function
ACR	absolute category rating
ACS	autocorrelation coefficient sequences
ACeS	Asia Cellular Satellite
ADC	analog-to-digital converter
ADPCM	adaptive differential pulse code modulation
AEC	acoustic echo cancelation
AFE	advanced front-end
AGC	automatic gain control
AGN	automatic gain normalization
AL	averaged acoustic frame likelihood
AMR-WB+	extended wide-band adaptive multirate coder
AMR-WB	wide-band AMR speech coder
AMSC-TMI	American Mobile Satellite Corporation Telesat Mobile Incorporated
AN	auditory nerve
ANC	active noise cancelation
ANN	artificial neural networks
ANOVA	analysis of variance
APA	affine projection algorithm
APC	adaptive predictive coding
APCO	Association of Public-Safety Communications Officials
APP	adjusted test-set perplexity
AR	autoregressive
ARISE	Automatic Railway Information Systems for Europe
ARMA	autoregressive moving-average
ARPA	Advanced Research Projects Agency
ARQ	automatic repeat request
ART	advanced recognition technology
ASAT	automatic speech attribute transcription
ASM	acoustic segment model
ASR	automatic speech recognition
ATF	acoustical transfer function
ATIS	airline travel information system
ATN	augmented transition networks
ATR	advanced telecommunications research
AW	acoustic word

B

BBN	Bolt, Beranek and Newman
BIC	Bayesian information criterion
BILD	binaural intelligibility level difference
BM	blocking matrix
BN	broadcast news
BSD	bark spectral distortion
BSS	blind source separation

C

C	consonants
CA	cochlear amplifier
CAF	continuous adaptation feedback
CART	classification and regression tree
CASA	computational auditory scene analysis
CAT	cluster adaptive training
CCR	comparison category rating
CDCN	codeword-dependent cepstral normalization
CDF	cumulative distribution function
CDMA	code division multiple access
CE	categorical estimation
CELP	code-excited linear prediction
CF	characteristic frequency
CF	coherence function
CH	call home
CHN	cepstral histogram normalization
CIS	caller identification system
CMLLR	constrained MLLR
CMOS	comparison mean opinion score
CMR	co-modulation masking release
CMS	cepstral mean subtraction
CMU	Carnegie Mellon University
CMVN	cepstral mean and variance normalization
CNG	comfort noise generation
COC	context-oriented clustering
CR	cross-relation
CRF	conditional random fields
CRLB	Cramèr–Rao lower bound
CS-ACELP	conjugate structure ACELP
CS-CELP	conjugate structure CELP
CSJ	corpus of spontaneous Japanese
CSR	continuous speech recognition
CTS	conversational telephone speech
CVC	consonant–vowel–consonant
CVN	cepstral variance normalization
CZT	chirp z-transform

D

DAC	digital-to-analog
DAG	directed acyclic graph
DAM	diagnostic acceptability measure

DARPA	Defense Advanced Research Projects Agency
DBN	dynamic Bayesian network
DCF	detection cost function
DCR	degradation category rating
DCT	discrete cosine transform
DET	detection error tradeoff
DF	disfluency
DFA	deterministic finite automata
DFT	discrete Fourier transform
DFW	dynamic frequency warping
DM	dialog management
DMOS	degradation mean opinion score
DP	dynamic programming
DPCM	differential PCM
DPMC	data-driven parallel model combination
DRT	diagnostic rhyme test
DSP	digital signal processing
DT	discriminative training
DTFT	discrete-time Fourier transform
DTW	dynamic time warping
DoD	Department of Defense

E

EC	equalization and cancelation
ECOC	error-correcting output coding
EER	equal error rate
EGG	electroglottography
EKF	extended Kalman filter
ELER	early-to-late energy ratio
EM	estimate–maximize
EM	expectation maximization
EMG	electromyographic
EMLLT	extended maximum likelihood linear transform
ER AAC-LD	error resilient low-delay advanced audio
ERB	equivalent rectangular bandwidth
ERL	echo return loss
ERLE	echo return loss enhancement
EVRC	enhanced variable rate coder
eX-CELP	extended CELP

F

FA	false accept
FAP	fast affine projection
FB-LPC	forward backward linear predictive coding
FBF	fixed beamformer
FBS	filter bank summation
FC	functional contour
FCDT	frame-count-dependent thresholding
FEC	frame erasure concealment
FFT	fast Fourier transform
FIFO	first-in first-out
FIR	finite impulse response
FM	forward masking
FMLLR	maximum-likelihood feature-space regression
FR	filler rate
FRC	functional residual capacity
FRLS	fast recursive least-squares
FSM	finite state machine
FSN	finite state network
FSS	frequency selective switch
FST	finite state transducer
FT	Fourier transform
FTF	fast transversal filter
FVQ	fuzzy vector quantization
FXLMS	filtered-X LMS

G

GCC	generalized cross-correlation
GCI	glottal closure instant
GEVD	generalized eigenvalue decomposition
GLDS	generalized linear discriminant sequence
GLR	generalized likelihood ratio
GMM	Gaussian mixture model
GPD	generalized probabilistic descent
GSC	generalized sidelobe canceller
GSM	Groupe Spéciale Mobile
GSV	GMM supervector

H

HLDA	heteroscedastic LDA
HLT	human language technologies
HMIHY	How May I Help You
HMM	hidden Markov models
HMP	hidden Markov processes
HNM	harmonic-plus-noise model
HOS	higher-order statistics
HPF	high-pass filter
HRTF	head-related transfer function
HSD	honestly significant difference
HSR	human speech recognition
HTK	hidden Markov model toolkit

I

IAI	International Association for Identification
ICA	independent component analysis
IDA	Institute for Defense Analyses
IDFT	inverse DFT
IDTFT	inverse discrete-time Fourier transform
IETF	Internet Engineering Task Force
IFSS	inverse frequency selective switch
IHC	inner hair cells
II	information index

IIR	infinite impulse response
iLBC	internet low-bit-rate codec
IMCRA	improved minima-controlled recursive averaging
IMDCT	inverse MDCT
IMM	interacting multiple model
IO	input–output
IP	internet protocol
IP	interruption (disfluent) point
IPA	International Phonetic Alphabet
IPNLMS	improved PNLMS
IQMF	QMF synthesis filterbank
IR	information retrieval
IS	Itakura–Saito
ISU	information state update
ITU	International Telecommunication Union
IVR	interactive voice response

J

JADE	joint approximate diagonalization of eigenmatrices
JND	just-noticeable difference

K

KL	Kullbach–Leibler
KLT	Karhunen–Loève transform

L

LAN	local-area network
LAR	log-area-ratio
LCMV	linearly constrained minimum-variance
LD-CELP	low-delay CELP
LDA	linear discriminant analysis
LDC	Linguistic Data Consortium
LDF	linear discriminant function
LID	language identification
LL	log-likelihood
LLAMA	learning library for large-margin classification
LLR	(log) likelihood ratio
LMFB	log mel-frequency filterbank
LMR	linear multivariate regression
LMS	least mean square
LNRE	large number of rare events
LP	linear prediction
LPC	linear prediction coefficients
LPC	linear predictive coding
LPCC	linear predictive cepstral coefficient
LRE	language recognition evaluation
LRT	likelihood-ratio test
LSA	latent semantic analysis
LSA	log-spectral amplitude
LSF	line spectral frequency
LSI	latent semantic indexing
LSR	low sampling rates
LTI	linear time invariant
LTIC	linear time-invariant causal
LTP	long term prediction
LVCSR	large vocabulary continuous speech recognition

M

M-step	maximization stage
MA	moving average
MAP	maximum a posteriori
MBE	multiband excited
MBR	minimum Bayes-risk
MBROLA	multiband resynthesis overlap-add
MC	multicategory
MCE	minimum classification error
MCN	multichannel Newton
MDC	multiple description coding
MDCT	modified discrete cosine transform
MDF	multidelay filter
MDP	Markov decision process
MELP	mixed excitation linear prediction
MFCC	mel-filter cepstral coefficient
MFoM	maximal figure-of-merit
MIMO	multiple-input multiple-output
MIPS	million instructions per second
ML	maximum-likelihood
MLLR	maximum-likelihood linear regression
MLP	multilayer perceptron
MMI	maximum mutual information
MMSE-LSA	MMSE of the log-spectral amplitude
MMSE-SA	MMSE of the spectral amplitude
MMSE	minimum mean-square error
MNB	measuring normalizing blocks
MNRU	modulated noise reference unit
MOPS	million operations per second
MOS	mean opinion score
MP	matching pursuit
MPE	minimum phone error
MPEG	Moving Pictures Expert Ggroup
MPI	minimal pairs intelligibility
MPLPC	multipulse linear predictive coding
MRI	magnetic resonance imaging
MRT	modified rhyme test
MS	minimum statistics
MSA	modern standard Arabic
MSD	minimum significant difference
MSE	mean-square error
MSG	maximum stable gain
MSNR	maximum signal-to-noise ratio
MSVQ	multistage VQ
MUI	multimodal user interface
MUSHRA	multi stimulus test with hidden reference and anchor

MVE	minimum verification error
MVIMP	my voice is my password
MW	maximum wins
MuSIC	multiple signal classification

N

NAB	North American Business News
NASA	National Aeronautics and Space Administration
NATO	North Atlantic Treaty Organization
NFA	nondeterministic finite automata
NFC	noise feedback coding
NIST	National Institute of Standards and Technology
NLG	natural language generation
NLMS	normalized least-mean-square
NLP	natural language processing
NLU	natural language understanding
NN	neural network
NSA	National Security Agency
NTT	Nippon Telephone & Telegraph
NUU	nonuniform units

O

O	object
OAE	otoacoustic emissions
OHC	outer-hair cells
OLA	overlap-and-add
OOV	out-of-vocabulary
OQ	open quotient
OR	out-of-vocabulary rate
OSI	open systems interconnection reference

P

P-PRLM	parallel PRLM
PARCOR	partial correlation coefficients
PBFD	partitioned-block frequency-domain
PCA	principal component analysis
PCBV	phonetic-class-based verification
PCFG	probabilistic context-free grammar
PCM	pulse-code modulation
PDA	pitch determination algorithms
PDC	personal digital cellular
PDF	probability density function
PDP	parallel distributed processing
PDS	positive-definite symmetric
PEAQ	perceptual quality assessment for digital audio
PESQ	perceptual evaluation of speech quality
PGG	photoglottography
PICOLA	pointer interval controlled overlap and add
PIV	particle image velocity
PLC	packet loss concealment
PLP	perceptual linear prediction
PMC	parallel model combination
PNLMS	proportionate NLMS
POS	part-of-speech
PP	word perplexity
PPRLM	parallel PRLM
PR	phone recognizer
PRA	partial-rank algorithm
PRLM	phoneme recognition followed by language modeling
PSD	power spectral density
PSI-CELP	pitch synchronous innovation CELP
PSI	pitch synchronous innovation
PSQM	perceptual speech quality measure
PSRELP	pitch-synchronous residual excited linear prediction
PSTN	public switched telephone network
PVT	parallel voice tokenization

Q

QA	question answering
QMF	quadrature mirror filter
QP	quadratic program
QoS	quality-of-service

R

RD	rate-distortion
RASTA	relative spectra
RATZ	multivariate Gaussian-based cepstral normalization
RCELP	relaxed CELP
REW	rapidly evolving waveform
RF	radio frequency
RIM	repair interval model
RIR	room impulse response
RL	reticular lamina
RLS	recursive least-squares
RM	resource management
RMS	root mean square
RMSE	root-mean-square error
ROC	receiver operating characteristic
RPA	raw phone accuracy
RPD	point of the reparandum
RPE-LTP	regular-pulse excitation with long-term prediction
RR	reprompt rates
RS	Reed–Solomon
RSVP	resource reservation protocol
RT	rich transcription

RTM	resonant tectorial membrane
RTP	real-time transport protocol

S

S2S	Speech-to-speech
S	subject
SAT	speaker-adaptive trained
SB	switchboard
SCTE	Society of Cable Telecommunications Engineers
SD-CFG	stochastic dependency context-free grammar
SD	spectral distortion
SDC	shifted delta cepstral
SDR-GSC	speech distortion regularized generalized sidelobe canceller
SDW-MWF	speech distortion-weighted multichannel Wiener filter
SEW	slowly evolving waveform
SGML	standard generalized markup language
SI	speech intelligibility
SII	speech intelligibility index
SIMO	single-input multiple-output
SISO	single-input single-output
SL	sensation level
SLM	statistical language model
SLS	spoken language system
SM	sinusoidal models
SMS	speaker model synthesis
SMT	statistical machine translation
SMV	selectable mode vocoder
SNR	signal-to-noise ratio
SOLA	synchronized overlap add
SOS	second-order statistics
SPAM	subspace-constrained precision and means
SPIN	saturated Poisson internal noise
SPINE	speech in noisy environment
SPL	sound pressure level
SPLICE	stereo piecewise linear compensation for environment
SQ	speed quotient
SR	speaking rate
SRT	speech reception threshold
SSML	speech synthesis markup language
STC	sinusoidal transform coder
STFT	short-time Fourier transform
SU	sentence unit
SUI	speech user interface
SUNDIAL	speech understanding and dialog
SUR	Speech Understanding Research
SVD	singular value decomposition
SVM	support vector machines
SWB	switchboard
SegSNR	segmental SNR

T

T–F	time–frequency
TBRR	transient beam-to-reference ratio
TC	text categorization
TCP	transmission control protocol
TCX	transform coded excitation
TD-PSOLA	time-domain pitch-synchronous overlap-add
TD	time domain
TDAC	time-domain aliasing cancelation
TDBWE	time-domain bandwidth extension
TDMA	time-division multiple-access
TDOA	time difference of arrival
TDT	topic detection and tracking
TF-GSC	transfer-function generalized sidelobe canceller
TFIDF	term frequency inverse document frequency
TFLLR	term frequency log-likelihood ratio
TFLOG	term frequency logarithmic
TI	transinformation index
TIA	Telecommunications Industry Association
TITO	two-input-two-output
TM	tectorial membrane
TM	tympanic membrane
TMJ	temporomandibular joint
TNS	temporal noise shaping
TPC	transform predictive coder
TSNFC	two-stage noise feedback coding
TTS	text-to-speech
ToBI	tone and break indices

U

UBM	universal background model
UCD	unit concatenative distortion
UDP	user datagram protocol
UE	user experience
UKF	unscented Kalman filter
ULD	ultra-low delay
USD	unit segmental distortion
USM	upward spread of masking
UT	unscented transform
UVT	universal voice tokenization

V

V	verb
V	vowels
VAD	voice activity detector
VBAP	vector based amplitude panning
VCV	vowel–consonant–vowel
VLSI	very large-scale integration

VMR-WB	variable-rate multimode wide-band
VOT	voice onset time
VQ	vector quantization
VRCP	voice recognition call processing
VRS	variable rate smoothing
VSC	vector space characterization
VSELP	vector sum excited linear prediction
VT	voice tokenization
VTLN	vocal-tract-length normalization
VTR	vocal tract resonance
VTS	vector Taylor-series
VoIP	voice over IP

W

WDRC	wide dynamic-range multiband compression
WER	word error rate
WFSA	weighted finite-state acceptors
WFST	weighted finite-state transducer
WGN	white Gaussian noise
WI	waveform interpolation
WLAN	wireless LAN
WLS	weighted least-squares
WMOPS	weighted MOPS
WSJ	Wall Street Journal
WSOLA	waveform similarity OLA
WiFi	wireless fidelity

X

XML	extensible mark-up languages
XOR	exclusive-or

Z

ZC	zero crossing
ZIR	zero-input response
ZSR	zero-state response

1. Introduction to Speech Processing

J. Benesty, M. M. Sondhi, Y. Huang

In this brief introduction we outline some major highlights in the history of speech processing. We briefly describe some of the important applications of speech processing. Finally, we introduce the reader to the various parts of this handbook.

1.1 A Brief History of Speech Processing

Human beings have long been motivated to create machines that can talk. Early attempts at understanding speech production consisted of building mechanical models to mimic the human vocal apparatus. Two such examples date back to the 13th century, when the German philosopher Albertus Magnus and the English scientist Roger Bacon are reputed to have constructed metal talking heads. However, no documentation of these devices is known to exist. The first documented attempts at making speaking machines came some five hundred years later. In 1769 Kratzenstein constructed resonant cavities which, when he excited them by a vibrating reed, produced the sounds of the five vowels a, e, i, o, and u. Around the same time, and independently of this work, Wolfgang von Kempelen constructed a mechanical speech synthesizer that could generate recognizable consonants, vowels, and some connected utterances. His book on his research, published in 1791, may be regarded as marking the beginnings of speech processing. Some 40 years later, Charles Wheatstone constructed a machine based essentially on von Kempelen's specifications [1.1–3].

Interest in mechanical analogs of the human vocal apparatus continued well into the 20th century. Mimics of the type of von Kempelen's machine were constructed by several people besides Wheatstone, e.g., Joseph Faber, Richard Paget, R. R. Riesz, et al.

It is known that as a young man Alexander Graham Bell had the opportunity to see Wheatstone's implementation. He too made a speaking machine of that general nature. However, it was his other invention – the telephone – that provided a major impetus to modern speech processing. Nobody could have guessed at that time the impact the telephone would have, not only on the way people communicate with each other but also on research in speech processing as a science in its own right. The availability of the speech waveform as an electrical signal shifted interest from mechanical to electrical machines for synthesizing and processing speech.

Some attempts were made in the 1920s and 1930s to synthesize speech electrically. However it is Homer Dudley's work in the 1930s that ushered in the modern era of speech processing. His most important contribution was the clear understanding of the *carrier* nature of speech [1.4]. He developed the analogy between speech signals and modulated-carrier radio signals that are used, for instance, for the transmission or broadcast of audio signals. In the case of the radio broadcast, the message to be transmitted is the audio signal which has frequencies in the range of 0–20 kHz. Analogously, the message to be transmitted in the case of speech is carried mainly by the time-varying shape of the vocal tract, which in turn is a representation of the *thoughts* the speaker wishes to convey to the listener. The movements of the vocal tract are at syllabic rates, i. e., at frequencies between 0 and 20 Hz. In each case – electromagnetic and acoustic – the message is in a frequency range unsuitable for transmission. The solution in each case is to imprint the message on a carrier. In the electromagnetic case the carrier is usually a high-frequency sinusoidal wave. In the acoustic case the carrier can be one of several signals. It is the quasi periodic signal provided by the vocal cords for voiced speech, and a noise-like signal provided by turbulence at a constriction for fricative and aspirated sounds. Or it can be a combination of these for voiced fricative sounds. Indeed, the selection of the carrier as well as the changes in intensity and fundamental frequency of the

vocal cords may be conveniently regarded as additional parts of the message.

Being an electrical engineer himself, Dudley proceeded to exploit this insight to construct an *electrical* speech synthesizer which dispensed with all the mechanical devices of von Kempelen's machine. Electrical circuits were used to generate the carriers. And the message (i. e., the characteristics of the vocal tract) was imprinted on the carrier by passing it through a time-varying filter whose frequency response was adjusted to simulate the transfer characteristics of the vocal tract.

With the collaboration of Riesz and Watkins, Dudley implemented two highly acclaimed devices based on this principle – the Voder and the Vocoder. The Voder was the first versatile talking machine able to produce arbitrary sentences. It was a system in which an operator manipulated a keyboard to control the sound source and the filter bank. This system was displayed with great success at the New York World Fair of 1939. It could produce speech of much better quality than had been possible with the mechanical devices, but remained essentially a curiosity. The Vocoder, on the other hand had a much more serious purpose. It was the first attempt at *compressing* speech. Dudley estimated that since the message in a speech signal is carried by the slowly time-varying filters, it should be possible to send adequate information for the receiver to be able to reconstruct a telephone speech signal using a bandwidth of only about 150 Hz, which is about 1/20 the bandwidth required to send the speech signal. Since bandwidth was very expensive in those days, this possibility was extremely attractive from a commercial point of view.

We have devoted so much space here to Dudley's work because his ideas were the basis of practically all the work on speech signal processing that followed. The description of speech in terms of a carrier (or excitation function) and its modulation (or the time-varying spectral envelope) is still – 70 years later – the basic representation. The parameters used to quantify these components, of course, have evolved in various ways. Besides the channel Vocoder (the modern name for Dudley's Vocoder) many other types of Vocoders have been invented, e.g., formant Vocoder, voice-excited Vocoder. Besides speech compression, Dudley's description was also considered for other applications such as secure voice systems, and the sound spectrograph and its use for communication with the deaf.

Unfortunately, the quality achieved by analog implementations of Vocoders never reached a level acceptable for commercial telephony. Nevertheless they found useful applications for military purposes where poor speech quality was tolerated. The Vocoder representation was also the basis of a speech secrecy system that found extensive use during World War II.

Another example of an analog implementation of Dudley's representation is the sound spectrograph. This is a device that displays the distribution of energy in a speech signal as a function of frequency, and the evolution of this distribution in time. This tool has been extremely useful for investigating properties of speech signals. A real time version of the spectrograph was intended for use as a device for communication with the deaf. That, however, was not very successful. A few people were able to identify about 300 words after 100 hours of training. However, it turned out to be too difficult a task to be practical.

During more than three decades following Dudley's pioneering work, a great amount of research was done on various aspects and properties of speech – properties of the speech production mechanisms, the auditory system, psychophysics, etc. However, except for the three applications mentioned above, little progress was made in speech signal processing and its applications. Exploitation of this research for practical applications had to wait for the general availability of digital hardware starting in the 1970s. Since then much progress has been made in speech coding for efficient transmission, speech synthesis, speech and speaker recognition, and hearing aids [1.5–7]. In the next section we discuss some of these developments.

Today, the area of speech processing is very vast and rich as can be seen from the contents of this Handbook. While we have made great progress since the invention of the telephone, research in the area of speech processing is still very active, and many challenging problems remain unsolved.

1.2 Applications of Speech Processing

As mentioned above, one of the earliest goals of speech processing was that of coding speech for efficient transmission. This was taken to be synonymous with reduction of the bandwidth required for transmitting speech. Several advances were needed before the modern success in speech coding was achieved. First, the

notions of information theory introduced during the late 1940s and 1950s brought the realization that the proper goal was the reduction of information rate rather than bandwidth. Second, hardware became available to utilize the sampling theorem to convert a continuous band-limited signal to a sequence of discrete samples. And quantization of the samples allowed digitization of a band-limited speech signal, thus making it usable for digital processing. Finally, the description of a speech signal in terms of linear prediction coefficients (LPC) provided a very convenient representation [1.8–11]. (The theory of predictive coding was in fact developed in 1955. However, its application to speech signals was not made until the late 1970s.)

A telephone speech signal, limited in frequency from 0 to 3.4 kHz, requires 64 kbps (kilobits per second) to be transmitted without further loss of quality. With modern speech compression techniques, the bit rate can be reduced to 13 kbps with little further degradation. For commercial telephony a remaining challenge is to reduce the required bit rate further but without sacrificing quality. Today, the rate can be lowered down to 2.4 kbps while maintaining very high intelligibility, but with a significant loss in quality. Some attempts have been made to reduce the bit rate down to 300 bps, e.g., for radio communication with a submarine. However the quality and intelligibility at these low bit rates are very poor.

Another highly successful application of speech processing is automatic speech recognition (ASR). Early attempts at ASR consisted of making deterministic models of whole words in a small vocabulary (say 100 words) and recognizing a given speech utterance as the word whose model comes closest to it. The introduction of hidden Markov models (HMMs) in the early 1980s provided a much more powerful tool for speech recognition [1.12–14]. Today many products have been developed that successfully utilize ASR for communication between humans and machines. And the recognition can be done for continuous speech using a large vocabulary, and in a speaker-independent manner. Performance of these devices, however, deteriorates in the presence of reverberation and even low levels of ambient noise. Robustness to noise, reverberation, and characteristics of the transducer, is still an unsolved problem.

The goal of ASR is to recognize speech accurately regardless of who the speaker is. The complementary problem is that of recognizing a speaker from his/her voice, regardless of what words he/she is speaking. At present this problem appears to be solvable only if the speaker is one of a small set of N known speakers. A variant of the problem is speaker *verification*, in which the aim is to automatically verify the claimed identity of a speaker. While speaker recognition requires the selection of one out of N possible outcomes, speaker verification requires just a yes/no answer. This problem can be solved with a high degree of accuracy for much larger populations. Speaker verification has application wherever access to data or facilities has to be controlled. Forensics is another area of application. The problem of reduced performance in the presence of noise, as mentioned above for ASR, applies also to speaker recognition and speaker verification.

A third application of speech processing is that of synthesizing speech corresponding to a given text. When used together with ASR, speech synthesis allows a complete two-way spoken interaction between humans and machines. Speech synthesis is also a way to communicate for persons unable to speak. Its use for this purpose by the famous physicist Stephen Hawking is well known.

Early attempts at speech synthesis consisted of deriving the time-varying spectrum for the sequence of phonemes of a given text sentence. From this the corresponding time variation of the vocal tract was estimated, and the speech was synthesized by exciting the time-varying vocal tract with periodic or noise-like excitation as appropriate. The quality of the synthesis was significantly improved by concatenating pre-stored units (i. e., short segments such as diphones, triphones) after modifying them to fit the context. Today the highest-quality speech is synthesized by the unit *selection* method in which the units are selected from a large amount of stored speech and concatenated with little or no modification.

Finally we might mention the application of speech processing to aids for the handicapped. Hearing aid technology has made considerable progress in the last two decades. Part of this progress is due to a slow but steady improvement in our knowledge of the human hearing mechanism. A large part is due to the availability of high-speed digital hardware. At present performance of hearing aids is still poor under noisy and reverberant conditions.

A potentially useful application of speech processing to aid the handicapped is to display the shape of one's vocal tract as one speaks. By trying to match one's vocal tract shape to a displayed shape, a deaf person can learn correct pronunciation. Some attempts to implement this idea have been made, but have still been only in the realm of research.

Another useful application is a reading aid for the blind. The idea is to have a device to scan printed text from a book, and synthesize speech from the

scanned text. Coupled with a device to change speaking rate, this forms a useful aid for the blind. Several products offering this application are available on the market.

Many other application examples are described in the various parts of this handbook. We invite the reader to browse this volume on speech processing to find topics relevant to his/her specific interests.

1.3 Organization of the Handbook

This handbook on speech processing is a comprehensive source of knowledge in speech technology and its applications. It is organized as follows. This volume is divided into nine parts. For each part we invited at least one associate editor (AE) to handle it. All the AEs are very well-known researchers in their respective area of research. Part A (AE: M. M. Sondhi) contains four chapters on production, perception, and modeling of speech signals. Part B (AEs: Y. Huang and J. Benesty) concerns signal processing tools for speech, in eight chapters. Part C (AE: B. Kleijn) covers five chapters on speech coding. In part D (AE: S. Narayanan), the areas of text-to-speech synthesis are presented in seven chapters. Part E (AEs: L. Rabiner and B.-H. Juang), with 10 chapters, is a comprehensive overview on speech recognition. Part F (AE: S. Parthasarathy) contains three chapters on speaker recognition. Part G (AE: C.-H. Lee) is about language identification and contains four chapters. In part H (AEs: J. Chen, S. Gannot, and J. Benesty), various aspects of speech enhancement are developed in seven chapters. Finally the last section, part I (AEs: J. Benesty, I. Cohen, and Y. Huang), presents the important aspects of multichannel speech processing in four chapters.

References

1.1 H. Dudley, T.H. Tarnoczy: The speaking machine of Wolfgang von Kempelen, J. Acoust. Soc. Am. **22**, 151–166 (1950)

1.2 G. Fant: *Acoustic Theory of Speech Production* (Mouton, 's-Gravenhage 1960)

1.3 J.L. Flanagan: *Speech Analysis, Synthesis and Perception* (Springer, New York 1972)

1.4 H. Dudley: The carrier nature of speech, Bell Syst. Tech. J. **19**(4), 495–515 (1940)

1.5 L.R. Rabiner, R.W. Schafer: *Digital Processing of Speech Signals* (Prentice Hall, Englewood Cliffs 1978)

1.6 S. Furui, M.M. Sondhi (Eds.): *Advances in Speech Signal Processing* (Marcel Dekker, New York 1992)

1.7 B. Gold, N. Morgan: *Speech and Audio Signal Processing* (Wiley, New York 2000)

1.8 P. Elias: Predictive coding I, IRE Trans. Inform. Theory **1**(1), 16–24 (1955)

1.9 P. Elias: Predictive coding II, IRE Trans. Inform. Theory **1**(1), 24–33 (1955)

1.10 B.S. Atal, M.R. Schroeder: Adaptive predictive coding of speech, Bell Syst. Tech. J. **49**(8), 1973–1986 (1970)

1.11 B.S. Atal: The history of linear prediction, IEEE Signal Proc. Mag. **23**(2), 154–161 (2006)

1.12 L.R. Rabiner: A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE **77**(2), 257–286 (1989)

1.13 L.R. Rabiner, B.-H. Juang: *Fundamentals of Speech Recognition* (Prentice-Hall, Englewood Cliff 1993)

1.14 F. Jelinek: *Statistical Methods for Speech Recognition* (MIT, Boston 1998)

3. Nonlinear Cochlear Signal Processing and Masking in Speech Perception

J. B. Allen

There are many classes of masking, but two major classes are easily defined: *neural masking* and *dynamic masking*. Neural masking characterizes the internal noise associated with the neural representation of the auditory signal, a form of loudness noise. Dynamic masking is strictly cochlear, and is associated with cochlear outer-hair-cell processing. This form is responsible for dynamic nonlinear cochlear gain changes associated with sensorineural hearing loss, the upward spread of masking, two-tone suppression and forward masking. The impact of these various forms of masking are critical to our understanding of speech and music processing. In this review, the details of what we know about nonlinear cochlear and basilar membrane signal processing is reviewed, and the implications of neural masking is modeled, with a comprehensive historical review of the masking literature. This review is appropriate for a series of graduate lectures on nonlinear cochlear speech and music processing, from an auditory point of view.

3.1 Basics

Auditory masking is critical to our understanding of speech and music processing. There are many classes of masking, but two major classes are easily defined. These two types of masking and their relation to nonlinear (NL) speech processing and coding are the focus of this chapter.

The *first* class of masking, denoted *neural masking*, is due to internal *neural noise*, characterized in terms of the intensity *just noticeable difference*, denoted $\Delta I(I, f, T)$ (abbreviated $\mathrm{JND_I}$) and defined as the *just discriminable change in intensity*. The $\mathrm{JND_I}$ is a function of intensity I, frequency f and stimulus type T (e.g., noise, tones, speech, music, etc.). As an *internal noise*, the $\mathrm{JND_I}$ may be modeled in terms of a loudness (i. e., perceptual intensity) noise density along the length of the cochlea ($0 \le X \le L$), described in terms of a *partial loudness JND* ($\Delta\mathcal{L}(X, T)$, a.k.a. $\mathrm{JND}_{\mathcal{L}}$). The cochlea or inner ear is the organ that converts signals from acoustical to neural signals. The loudness JND is a function of the *partial loudness* $\mathcal{L}(X)$, defined as the loudness contribution coming from each cochlear *critical band*, or more generally, along some *tonotopic central auditory representation*. The critical band is a measure of cochlear bandwidth at a given cochlear *place* X. The loudness JND plays a major role in speech and music coding since coding quantization noise may be masked by this internal quantization (i. e., *loudness noise*).

The *second* masking class, denoted here as *dynamic masking*, comes from the NL mechanical action of cochlear *outer-hair-cell* (OHC) signal processing. It can have two forms, simultaneous and nonsimultaneous, also known as *forward masking*, or *post-masking*. Dynamic-masking (i. e., nonlinear OHC signal processing) is well known (i. e., there is a historical literature

on this topic) to be intimately related to questions of cochlear frequency selectivity, sensitivity, dynamic range compression and *loudness recruitment* (the loss of loudness dynamic range). Dynamic masking includes the *upward spread of masking* (USM) effect, or in neural processing parlance, *two-tone suppression* (2TS). It may be underappreciated that NL OHC processing (i. e., dynamic masking) is largely responsible for *forward masking* (FM, or post-stimulus masking), which shows large effects over long time scales. For example OHC effects (FM/USM/2TS) can be as large as 50 dB, with an FM *latency* (return to base line) of up to 200 ms. *Forward masking* (FM) and NL OHC *signal onset enhancement* are important to the detection and identification of perceptual features of a speech signal. Some research has concluded that forward masking is not related to OHC processing [3.1, 2], so the topic remains controversial. Understanding and modeling NL OHC processing is key to many speech processing applications. As a result, a vibrant research effort driven by the National Institute of Health on OHC biophysics has ensued.

This OHC research effort is paying off at the highest level. Three key examples are notable. *First* is the development of wide dynamic-range multiband compression (WDRC) hearing aids. In the last 10–15 years WDRC signal processing (first proposed in 1937 by researchers at Bell Labs [3.3]), revolutionized the hearing-aid industry. With the introduction of compression signal processing, hearing aids now address the recruitment problem, thereby providing speech audibility over a much larger dynamic range, at least in quiet. The problems of the impaired ear given speech in noise is poorly understood today, but this problem is likely related to the effects of NL OHC processing. This powerful circuit (WDRC) is not the only reason hearing aids of today are better. Improved electronics and transducers have made significant strides as well. In the last few years the digital barrier has finally been broken, with digital signal processing hearing aids now becoming common.

A *second* example is the development of otoacoustic emissions (OAE) as a hearing diagnostic tool. Pioneered by David Kemp and Duck Kim, and then developed by many others, this tool allows for cochlear evaluation of neonates. The identification of cochlear hearing loss in the first month has dramatically improves the lives of these children (and their parents). While it is tragic to be born deaf, it is much more tragic for the deafness to go unrecognized until the child is three years old, when they fail to learn to talk. If you cannot hear you do not learn to talk. With proper and early cochlear implant intervention, these kids can lead nearly normal-hearing lives and even talk on the phone. However they cannot understand speech in noise. It is at least possible that this loss is due to the lack of NL OHC processing.

A *third* example of the application of NL OHC processing to speech processing is still an underdeveloped application area. The key open problem here is: *How does the auditory system, including the NL cochlea, followed by the auditory cortex, processes human speech*? There are many aspects of this problem including speech coding, speech recognition in noise, hearing aids and language learning and reading disorders in children. If we can solve the *robust phone decoding problem*, we will fundamentally change the effectiveness of human-machine interactions. For example, the ultimate hearing aid is the hearing aid with built in robust speech feature detection and phone recognition. While we have no idea when this will come to be, and it is undoubtedly many years off, when it happens there will be a technology revolution that will change human communications.

In this chapter several topics will be reviewed. First is the history of cochlear models including extensions that have taken place in recent years. These models include both macromechanics and micromechanics of the tectorial membrane and hair cells. This leads to comparisons of the basilar membrane, hair cell, and neural frequency tuning. Hearing loss, loudness recruitment, as well as other key topics of modern hearing health care, are discussed. The role of NL mechanics and dynamic range are reviewed to help the reader understand the importance of modern wideband dynamic range compression hearing aids as well as the overall impact of NL OHC processing.

Any reader desiring further knowledge about cochlear anatomy and function or a basic description of hearing, they may consult *Pickles* [3.4], *Dallos* [3.5], *Yost* [3.6].

3.1.1 Function of the Inner Ear

The goal of cochlear modeling is to refine our understanding of how auditory signals are processed. The two main roles of the cochlea are to separate the input acoustic signal into overlapping frequency bands, and to compress the large acoustic intensity range into the much smaller mechanical and electrical dynamic range of the inner hair cell. This is a basic question of information processing by the ear. The eye plays a similar role as a peripheral organ. It breaks the light image into rod- and cone-sized pixels, as it compresses the dynamic range of the visual signal. Based on the intensity JND, the corresponding visual dynamic range is about nine to

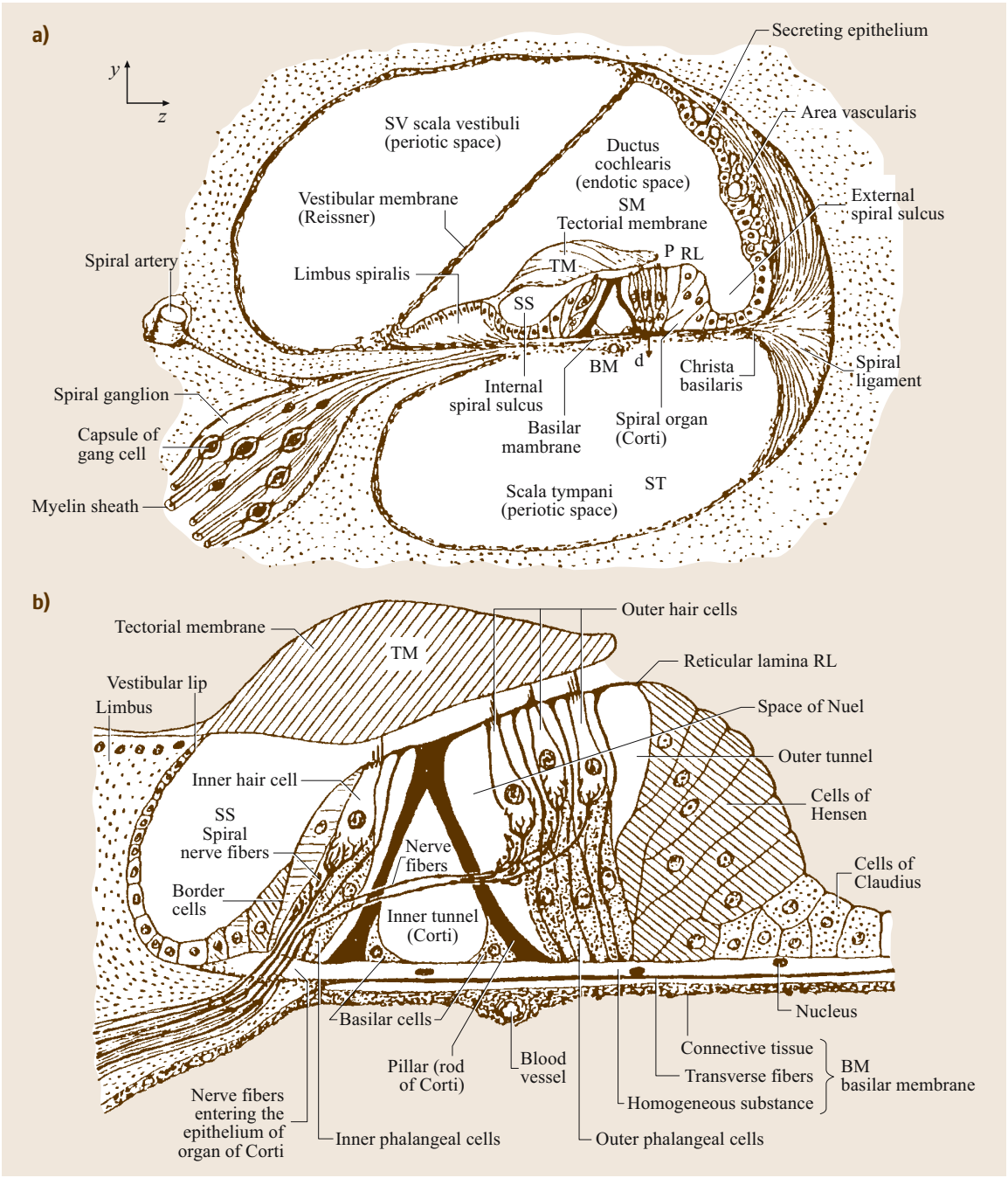


Fig. 3.1a,b On the left we see all the major structures of the cochlea **(a)**. The three chambers are filled with fluid. Reissner's membrane is an electrical barrier and is not believed to play a mechanical role. The right panel **(b)** shows the inner and outer hair cells, pillar cells and other supporting structures, the basilar membrane (BM), and the tectorial membrane (TM)

ten orders of magnitude of intensity [3.7, 8], while the ear has about 11 to 12. The stimulus has a relatively high information rate. Neurons are low-bandwidth channels. The eye and the ear must cope with this problem by reducing the stimulus to a large number of low bandwidth signals. It is then the job of the cortex to piece these pixel signals back together, to reconstruct the world as we see and hear it.

The acoustic information coding starts in the cochlea (Fig. 3.1a) which is composed of three major chambers formed by Reissner's membrane and the basilar membrane (BM). Mechanically speaking, there are only two chambers, as Reissner's membrane is only for electrical isolation of the scala media (SM) [3.4, 5]. Figure 3.1b shows a blown-up view of the organ of Corti where the inner hair cells (IHC) and outer hair cells (OHC) sit between the BM and the tectorial membrane (TM). As the BM moves up and down, the TM shears against the reticular lamina (RL), causing the cilia of the inner and outer hair cells to bend. The afferent auditory nerve fibers that are connected to the inner hair cells carry the signal information into the auditory system. Many fewer efferent fibers bring signals from the auditory system to the base of the outer hair cells. The exact purpose of these efferent fibers remains unknown.

Inner Hair Cells

In very general terms, the role of the cochlea is to convert sound at the eardrum into neural pulse patterns along approximately 30 000 neurons of the human auditory (VIIIth) nerve. After being filtered by the cochlea, a low-level pure tone has a narrow spread of excitation which excites the cilia of about 40 contiguous inner hair cells [3.5, 9, 10]. The IHC excitation signal has a narrow bandwidth and a center frequency that depends on the inner hair cell's location along the basilar membrane. Each hair cell is about 10 μm in diameter while the human basilar membrane is about 35 mm in length (35 000 μm). Thus the neurons of the auditory nerve encode the responses of about 3500 inner hair cells which form a single row of cells along the length of the BM. Each inner-hair-cell voltage is a low-pass-filtered representation of the detected inner-hair-cell cilia displacement [3.11]. Each hair cell is connected to many neurons, having a wide range of spontaneous firing rates and thresholds [3.12]. In the cat, for example, approximately 15–20 neurons encode each of these narrow band inner hair cells with a neural timing code. It is commonly accepted that all mammalian cochleae are similar in function except the frequency range of operation differs between species (e.g., human $\approx 0.1-20$ kHz and cat $\approx 0.3-50$ kHz). It is widely believed that the neuron information channel between the hair cell and the *cochlear nucleus* is a combination of the mean firing rate and the relative timing between neural pulses (spikes). The mean firing rate is reflected in the loudness coding, while the relative timing carries more subtle cues, including for example pitch information such as speech voicing distinctions.

Outer Hair Cells

As shown in Fig. 3.1b there are typically three (occasionally four) outer hair cells (OHCs) for each inner hair cell (IHCs), leading to approximately 12 000 OHCs in the human cochlea. Outer hair cells are used for intensity dynamic-range control. This is a form of NL signal processing, not dissimilar to Dolby sound processing. This form of processing was inspired by cochlear function, and was in use long before it was patented by Dolby, in movie sound systems developed by Bell Labs in the 1930s and 1940s. Telephone speech is similarly compressed [3.13] via μ-law coding. It is well known (as was first proposed by *Lorente de Nó* [3.14] and *Steinberg* [3.3]) that noise damage of *nerve cells* (i. e., OHCs) leads to a reduction of dynamic range, a disorder clinically named *loudness recruitment*. The word *recruitment*, which describes the abnormal growth of loudness in the impaired ear, is a seriously misleading term, since nothing is being recruited [3.15].

We may describe cochlear processing two ways: first in terms of the signal representation at various points in the system; and second, in terms of models which are our most succinct means of conveying the conclusions of years of detailed and difficult experimental work on cochlear function. The body of experimental knowledge has been very efficiently represented (to the extent that it is understood) in the form of these mathematical models. When no model exists (e.g., because we do not understand the function), a more basic description via the experimental data is necessary. Several good books and review papers that make excellent supplemental reading are available [3.4, 8, 16, 17].

For pedagogical purposes this chapter has been divided into four parts. Besides this introduction, we include sections on the NL cochlea, neural masking, and finally a brief discussion. Section 3.2 discusses dynamic masking due to NL aspects of the cochlear outer hair cells. This includes the practical aspects, and theory, of the upward spread of masking (USM) and two-tone suppression. Section 3.3 discusses neural masking, the JND, loudness recruitment, the loudness signal-to-noise ratio

(SNR), and the Weber fraction. Section 3.4 provides a brief summary.

3.1.2 History of Cochlear Modeling

Typically the cochlea is treated as an uncoiled long thin box, as shown in Fig. 3.2a. This represents the starting point for the macromechanical models.

Macromechanics

In his book *On the Sensations of Tone Helmholtz* [3.18] likened the cochlea to a bank of highly tuned resonators selective to different frequencies, much like a piano or a harp [3.19, p. 22–58], with each string representing a different place X on the basilar membrane. This model as proposed was quite limited since it leaves out key features, the most important of which is the cochlear fluid coupling between the mechanical resonators. But given the early publication date, the great master of physics and psychophysics Helmholtz shows deep insight and his studies provided many very important contributions.

The next major contribution by *Wegel* and *Lane* [3.20] stands in a class of its own even today, as a double-barreled paper having both deep psychophysical and modeling insight. Fletcher published much of the Wegel and Lane data one year earlier [3.21]. It is not clear to me why Wegel and Lane are always quoted for these results rather than Fletcher. In Fletcher's 1930 modeling paper, he mentioned that he was the subject in the Wegel and Lane study. It seems to me that Fletcher deserves some of the credit. The paper was the first to quantitatively describe the details of how a high level low frequency tone affects the audibility of a second low-level higher-frequency tone (i. e., the *upward spread of masking*). It was also the first publication to propose a *modern* model of the cochlea, as shown in Fig. 3.2b. If Wegel and Lane had been able to solve the model equations implied by their circuit (of course they had no computer to do this), they would have predicted cochlear traveling waves. It was their mistake, in my opinion, to make this a single paper. The modeling portion of their paper has been totally overshadowed by their experimental results. Transmission line theory had been widely exploited by Campbell, the first mathematical research at AT&T research (ca. 1898) with the invention of the wave filter [3.22, 23], which had been used for speech articulation studies [3.24–26], and Fletcher and Wegel were fully utilizing Campbell's important discoveries.

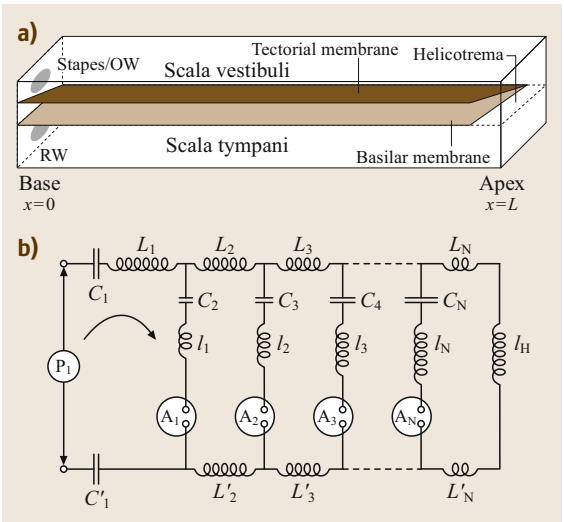


Fig. 3.2a,b On the left (a) see the basic 2-D box model of the cochlea. The *Base* ($x = 0$) is the high-frequency end of the cochlea while the *Apex* ($x = L$) carries the low frequencies. On the right (b) the 1924 Wegel and Lane electrical equivalent circuit. The model is built from a cascade of electrical sections

It was the experimental observations of G. von Békésy starting in 1928 on human cadaver cochleae which unveiled the physical nature of the basilar membrane traveling wave. What von Békésy found (consistent with the 1924 Wegel and Lane model) was that the cochlea is analogous to a *dispersive* transmission line where the different frequency components which make up the input signal travel at different speeds along the basilar membrane, thereby isolating each frequency component at a different place X along the basilar membrane. He properly identified this dispersive wave a *traveling wave,* just as Wegel and Lane had predicted in their 1924 model of the cochlea.

Over the intervening years these experiments have been greatly improved, but von Békésy's fundamental observation of the traveling wave still stands. His original experimental results, however, are *not* characteristic of the responses seen in more-recent experiments, in many important ways. These differences are believed to be due to the fact that Békésy's cochleae were dead, and because of the high sound levels his experiments required. He observed the traveling wave using stroboscopic light, in dead human cochleae, at sound levels well above 140 dB – SPL.

Today we find that for a pure tone input the traveling wave has a more sharply defined location on the basilar membrane than that observed by von Békésy. In fact, according to measurements made over the last 20 years, the response of the basilar membrane to a pure tone

can change in amplitude by more than five orders of magnitude per millimeter of distance along the basilar membrane (e.g., 300 dB/oct is equivalent to 100 dB/mm in the cat cochlea).

The One-Dimensional Model of the Cochlea

To describe this response it is helpful to call upon the macromechanical *transmission line models* of *Wegel* [3.20] (Fig. 3.2b) and *Fletcher* [3.27], first quantitatively analyzed by *Zwislocki* [3.28, 29], *Ranke* [3.30], *Peterson* and *Bogert* [3.31], *Fletcher* [3.32, 33]. This popular transmission line model is now denoted the *one-dimensional* (1-D), or *long-wave* model.

Zwislocki [3.28] was first to quantitatively analyze Wegel and Lane's macromechanical cochlear model, explaining Békésy's traveling wave observations. The stapes input pressure P_1 is at the left, with the input velocity V_1, as shown by the arrow, corresponding to the stapes velocity. This model represents the mass of the fluids of the cochlea as electrical inductors and the BM stiffness as capacitors. Electrical circuit networks are useful when describing mechanical systems. This is possible because of an electrical to mechanical analog that relates the two systems of equations. Electrical circuit elements comprise a de facto standard for describing such equations. It is possible to write down the equations that describe the system from the circuit of Fig. 3.2b, by those trained in the art. Engineers and scientists frequently find it easier to *read* and think in terms of these pictorial circuit diagrams, than to interpret the corresponding equations.

BM Impedance. During the following discussion it is necessary to introduce the concept of a *one-port* (two-wire) impedance. *Ohm's law* defines the impedance as

$$\text{Impedance} = \frac{\text{effort}}{\text{flow}} \,. \tag{3.1}$$

In an electrical system the impedance is the ratio of a voltage (effort) over a current (flow). In a mechanical system it is the force (effort) over the velocity (flow).

For *linear time-invariant causal* (LTIC) systems (i. e., an impedance), *phasor* notation is very useful, where the tone is represented as the real part (Re) of the complex exponential

$$\mathrm{e}^{\mathrm{i}2\pi ft+\mathrm{i}\phi} \equiv \cos(2\pi ft+\phi) + \mathrm{i}\sin(2\pi ft+\phi) \,. \tag{3.2}$$

The symbol $\equiv$ denotes *equivalence*. It means that the quantity to the left of $\equiv$ is defined by the quantity on the right. More specifically, impedance is typically defined in the frequency domain using *Laplace transform* notation, in terms of a damped tone

$$A\,\mathrm{e}^{\sigma t}\cos(2\pi ft+\phi) \equiv A\,\mathrm{Re}\,\mathrm{e}^{st+\mathrm{i}\phi} \tag{3.3}$$

excitation, characterized by the tone's amplitude A, phase ϕ and *complex Laplace frequency* $s \equiv \sigma + \mathrm{i}2\pi f$. When a function such as $Z(s)$ is shown as a function of the complex frequency s, this means that its inverse Laplace transform $z(t) \leftrightarrow Z(s)$ must be *causal*. In the time domain, the voltage may be found from the current via a convolution with $z(t)$. Three classic examples of such impedances are presented next.

Example 3.1: The impedance of the tympanic membrane (TM, or eardrum) is defined in terms of a pure tone pressure in the ear canal divided by the resulting TM volume velocity (the velocity times the area of TM motion) [3.34, 35]. The pressure (effort) and volume velocity (flow) referred to here are conventionally described using complex numbers, to account for the phase relationship between the two.

Example 3.2: The impedance of a spring is given by the ratio of the force $F(f)$ to velocity $V(f) = sX(f)$ with displacement X

$$Z(s) \equiv \frac{F}{V} = \frac{K}{s} = \frac{1}{sC} \,, \tag{3.4}$$

where the spring constant K is the stiffness, C the compliance, and s is the complex radian frequency. The stiffness is represented electrically as a capacitor (as parallel lines in Fig. 3.2b). Having $s = \sigma + \mathrm{i}2\pi f$ in the denominator indicates that the impedance of a spring has a phase of $-\pi/2$ (e.g., $-90°$). Such a phase means that when the velocity is $\cos(2\pi ft)$, the force is $\sin(2\pi ft)$. This follows from Hooke's law

$$F = KX = \frac{K}{s}sX = \frac{K}{s}V \,. \tag{3.5}$$

Example 3.3: From *Newton's law* $F = Ma$ where F is the force, M is the mass, and acceleration $a(s) = sV(s)$ (i. e., the acceleration in the time domain is $\mathrm{d}v(t)/\mathrm{d}t$). The electrical element corresponding to a mass is an *inductor,* indicated in Fig. 3.2b by a coil. Thus for a mass $Z(s) = sM$.

From these relations the magnitude of the impedance of a spring decreases as $1/f$, while the impedance magnitude of a mass is proportional to f. The stiffness with

its −90° phase is called a *lagging* phase, while the mass with its +90° phase is called a *leading* phase.

Different points along the basilar membrane are represented by the cascaded sections of the lumped transmission line model of Fig. 3.2b. The position X along the model is called the *place* variable and corresponds to the longitudinal position along the cochlea. The series (horizontal) inductors (coils) denoted by L_k represent the fluid mass (inertia) along the length of the cochlea, while the shunt elements represent the mechanical (acoustical) impedance of the corresponding partition (organ of Corti) impedance, defined as the pressure drop across the partition divided by its volume velocity per unit length

$$Z_{\mathrm{p}}(s, X) = \frac{K_{\mathrm{p}}(X)}{s} + R_{\mathrm{p}}(X) + sM_{\mathrm{p}} \,, \tag{3.6}$$

where $K(X)$ is the partition stiffness, and R_{p} is the partition resistance. Each inductor going to ground (l_i in Fig. 3.2b) represents the partition plus fluid mass per unit length M_{p} of the section. Note that sM, R_{p} and K/s are impedances, but the mass M and stiffness K are not. The partition stiffness decreases exponentially along the length of the cochlea, while the mass is frequently approximated as being independent of place.

As shown in Fig. 3.3a, for a given input frequency the BM impedance magnitude has a local minimum at the shunt resonant frequency, where the membrane that can move in a relatively unrestricted manner. The shunt *resonance* has special significance because at this resonance frequency $F_{\mathrm{cf}}(X)$ the inductor and the capacitor reactance cancel each other, creating an acoustic *hole*, where the only impedance element that contributes to the flow resistance is R_{p}. Solving for $F_{\mathrm{cf}}(X)$

$$\frac{K_{\mathrm{p}}(X)}{2\pi \mathrm{i} F_{\mathrm{cf}}} + 2\pi \mathrm{i} F_{\mathrm{cf}} M_{\mathrm{p}} = 0 \tag{3.7}$$

defines the *cochlear map function*, which is a key concept in cochlear modeling:

$$F_{\mathrm{cf}}(X) \equiv \frac{1}{2\pi}\sqrt{\frac{K_{\mathrm{p}}(X)}{M_{\mathrm{p}}}} \,. \tag{3.8}$$

The inverse of this function specifies the location of the *hole* $X_{\mathrm{cf}}(f)$ as shown in Fig. 3.3a. In the example of Fig. 3.3a two frequencies are show, at 1 and 8 kHz, with corresponding resonant points shown by $X_{\mathrm{cf}}(1)$ and $X_{\mathrm{cf}}(8)$.

Basal to $X_{\mathrm{cf}}(f)$ in Fig. 3.3a, the basilar membrane is increasingly stiff, and apically (to the right of the resonant point), the impedance is mass dominated. The above description is dependent on the input frequency f since the location of the hole is frequency dependent. In this apical region the impedance has little influence since almost no fluid flows past the low-impedance hole. This description is key to our understanding of why the various frequency components of a signal are splayed out along the basilar membrane.

If one puts a pulse of current in at the stapes, the highest frequencies that make up the pulse would be shunted close to the stapes since at high frequencies the hole is near the stapes, while the lower frequencies would continue down the line. As the low-pass pulse travels down the basilar membrane, the higher frequencies are progressively removed, until almost nothing is left when the pulse reaches the end of the model (the helicotrema end, the apex of the cochlea).

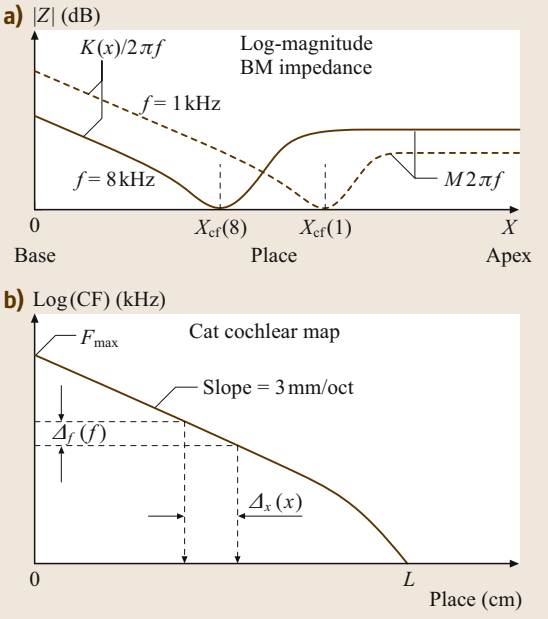


Fig. 3.3 **(a)** Plot of the log-magnitude of the impedance as a function of place for two different frequencies of 1 and 8 kHz showing the impedance; the region labeled $K(X)$ is the region dominated by the stiffness and has impedance $K(X)/s$. The region labeled M is dominated by the mass and has impedance sM. The characteristic places for 1 and 8 kHz are shown as X_{cf}. **(b)** Cochlear map of the cat following Liberman and Dodds. The resonance frequency depends on place according to the *cochlear map function* **(b)**. A *critical bandwidth* $\Delta_f(f)$ and a *critical spread* $\Delta_x(X)$ area related through the cochlear map

When a single tone is played, the response in the base increases in proportion to the BM compliance (inversely with the stiffness) until there is is a local maximum just before the traveling wave reaches the resonant hole, at which point the response plummets, since the fluid flow is shorted by the hole. For a fixed stimulus frequency f there is a maximum along the place axis called the *characteristic place*, denoted by $X_{\mathrm{cf}}^{(p)}(f)$. Likewise at a given place X as a function of frequency there is a local maximum called the *characteristic frequency*, denoted by $F_{\mathrm{cf}}^{(p)}(X)$. The relation between the peak in place as a function of frequency or of the peak in frequency as a function of place is also called the *cochlear map*. There is serious confusion with conventional terminology here. The resonant frequency of the BM impedance mathematically defines F_{cf} and specifies the frequency on the base of the high-frequency steep portion of the tuning slope, *not* the peak. However the peak is used as the visual cue, *not* the base of the high-frequency slope. These two definitions differ by a small factor (that is ignored) that depends directly on the high-frequency slope of the response. Over most of the frequency range this slope is huge, resulting in a very small factor, justifying its being ignored. However at very low frequencies the slope is shallow and the factor can then be large. The *droop* in the cochlear map seen in Fig. 3.3b at the apex ($x = L$) may be a result of these conflicting definitions. The cochlear map function $F_{\mathrm{cf}}(X)$ plays a key role in cochlear mechanics, has a long history, and is known by many names [3.27, 36–40], the most common today being *Greenwood's function*. In the speech literature it is called the *Mel scale*.

The spread of the response around the peak for a fixed frequency is denoted the *critical spread* $\Delta_x(f)$, while the frequency spread at a given place is called the *critical band* denoted $\Delta_f(X)$. As early as 1933 it was clear that the critical band must exist, as extensively discussed by *Fletcher* and *Munson* [3.41]. At any point along the BM the critical band is proportional to the *critical ratio* $\kappa(X)$, defined as the ratio of pure tone detection intensity at threshold in a background of white noise, to the spectral level of the noise [3.42], namely

$$\Delta_f(X) \propto \kappa(X) \,. \tag{3.9}$$

In the next section we shall show how the the relations between these various quantities are related via the cochlear map.

Derivation of the Cochlear Map Function. The derivation of the cochlear map is based on *counting* critical bands as shown by *Fletcher* [3.10] and popularized by *Greenwood* [3.43]. The *number of critical bands* N_{cb} may be found by integrating the critical band density over both frequency and place, and equating these two integrals, resulting in the cochlear map $F_{\mathrm{cf}}(X)$:

$$N_{\mathrm{cb}} \equiv \int_0^{X_{\mathrm{cf}}} \frac{\mathrm{d}X}{\Delta_x(X)} = \int_0^{F_{\mathrm{cf}}} \frac{\mathrm{d}f}{\Delta_f(f)} \,. \tag{3.10}$$

There are approximately 20 pure-tone frequency JNDs per critical band [3.37], [3.42, p. 171], and *Fletcher* showed that the *critical ratio* expressed in dB $\kappa_{\mathrm{dB}}(X)$ is of the form $aX + b$, where a and b are constants [3.10]. As verified by *Greenwood* [3.43, p. 1350, (1)] the critical bandwidth in Hz is therefore

$$\Delta_f(X) \propto 10^{\kappa_{\mathrm{dB}}(X)/10} \,. \tag{3.11}$$

The critical spread $\Delta_x(X)$ is the effective width of the energy spread on the basilar membrane for a pure tone. Based on a suggestion by Fletcher, *Allen* showed that for the cat, $\Delta_x(X)$ corresponds to about 2.75 times the basilar membrane width with $W_{\mathrm{bm}}(X) \propto \mathrm{e}^X$ [3.10]. It is reasonable to assume that the same relation would hold in the human case.

The direct observation of the cochlear map in the cat was made by *Liberman* [3.44] and *Liberman* and *Dodds* [3.45], and they showed the following empirical formula fit the data

$$F_{\mathrm{cf}}(X) = 456(10^{2.1(1-X/L)} - 0.8) \,, \tag{3.12}$$

where the length of the cat cochlea is $L = 21$ mm, and X is measured from the stapes [3.44]. The same formula may be used for the human cochlea if $L = 35$ mm is used, the 456 is replaced by 165.4, and 0.8 by 0.88. Based on (3.12), and as defined in Fig. 3.3b, the *slope* of the cochlear map is 3 mm/oct for the cat and 5 mm/oct for the human, as may be determined from the formula $L \log_{10}(2)/2.1$ with $L = 21$ or 35 for cat and human, respectively.

For a discussion of work after 1960 on the critical band see *Allen* [3.10] and *Hartmann* [3.17].

3.2 The Nonlinear Cochlea

3.2.1 Cochlear Modeling

In cochlear modeling there are two fundamental intertwined complex problems, *cochlear frequency selectivity* and *cochlear/OHC nonlinearity*. Wegel and Lane's 1924 transmission line wave theory was a most important development, since it was published 26 years prior to the experimental results of von Békésy, and it was based on a simple set of physical principles, conservation of fluid mass, and a spatially variable basilar membrane stiffness. It gives insight into both the NL cochlea, as well has two-dimensional (2-D) model frequency-selective wave-transmission effects (mass loading of the BM).

Over a 15 year period starting in 1971, there was a paradigm shift. Three discoveries rocked the field:

1. nonlinear compressive basilar membrane and inner-hair-cell measures of neural-like cochlear frequency selectivity [3.47, 48],
2. otoacoustic (ear canal) nonlinear emissions [3.49], and
3. motile outer hair cells [3.50].

Today we know that these observations are related, and all involve outer hair cells. A theory (e.g., a computational model) is needed to tie these results together. Many groups are presently working out such theories.

On the modeling side during the same period (the 1970's) all the variants of Wegel and Lane 1-D linear theory were becoming dated because:

1. numerical model results became available, which showed that 2- and three-dimensional (3-D) models were more frequency selective than the 1-D model,
2. experimental basilar membrane observations showed that the basilar membrane motion had a nonlinear compressive response growth, and
3. improved experimental basilar membrane observations became available which showed increased nonlinear cochlear frequency selectivity.

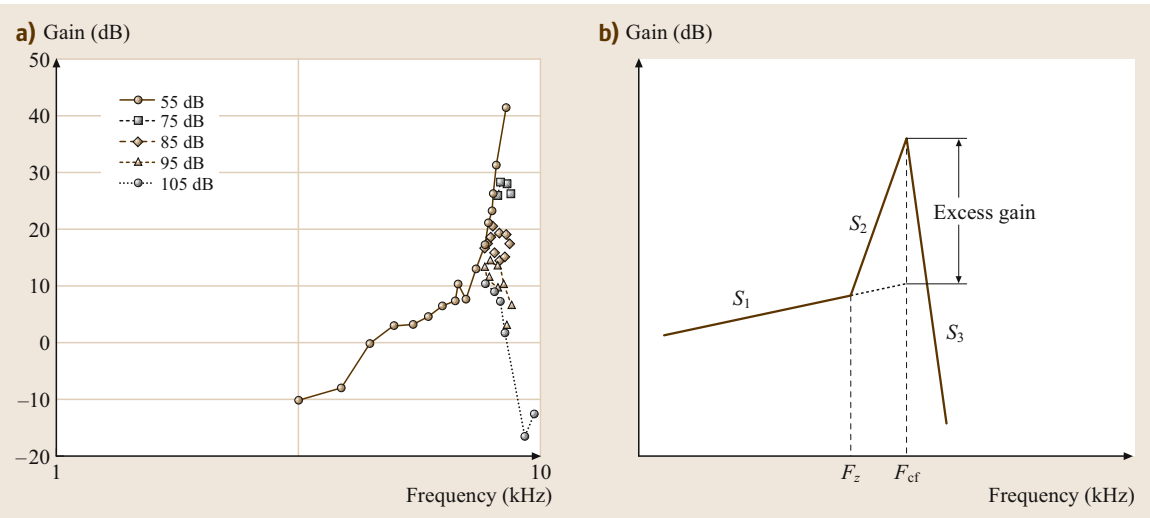


Fig. 3.4a,b There are six numbers that characterize every curve, three slopes (S_1, S_2, S_3), in dB/oct, two frequencies (F_z, F_{cf}), and the *excess gain* characterizes the amount of gain at F_{cf} relative to the gain defined by S_1. The excess gain depends on the input level for the case of a nonlinear response like the cochlea. Rhode found up to ≈ 35 dB of excess gain at 7.4 kHz and 55 dB − SPL, relative to the gain at 105 dB − SPL. From of the 55 dB − SPL curve of **(a)** (the most sensitive case), and his Table I, $S_1 = 9$, $S_2 = 86$, and $S_3 = -288$ (dB/oct), $F_z = 5$ kHz, $F_{cf} = 7.4$ kHz, and an *excess gain* of 27 dB. Rhode reported $S_1 = 6$ dB/oct, but 9 seems to be a better fit to the data, so 9 dB/oct is the value we have used for our comparisons. **(a)** Response of the basilar membrane for his most sensitive animal. The graduations along the abscissa are at 0.1, 1.0 and 10.0 kHz (after [3.46, Fig. 9a]) **(b)** Basic definition of the 6 parameters for characterizing a tuning curve: slopes S_1, S_2, S_3, frequencies F_z and F_{cf}, and the excess gain

Because these models and measures are still under development today [the problem has not yet (ca. 2007) been solved], it is necessary to describe the data rather than the models. Data that drives these nonlinear cochlear measures include:

- The upward spread of masking (USM), first described quantitatively by Wegel and Lane in 1924,
- Distortion components generated by the cochlea and described by *Wegel* and *Lane* [3.20], *Goldstein* and *Kiang* [3.52], *Smoorenburg* [3.53], *Kemp* [3.54], *Kim* et al. [3.55], *Fahey* and *Allen* [3.56] and many others,
- Normal loudness growth and recruitment in the impaired ear [3.3, 41],
- The frequency dependent neural two-tone suppression observed by *Sachs* and *Kiang* [3.57], *Arthur* et al. [3.58], *Kiang* and *Moxon* [3.59], *Abbas* and *Sachs* [3.60], *Fahey* and *Allen* [3.56], *Pang* and *Guinan* [3.61], and others,
- The frequency-dependent basilar membrane response-level compression first described by *Rhode* [3.46, 47],
- The frequency-dependent inner-hair-cell receptor potential level compression, first described by *Sellick* and *Russell* [3.48], *Russell* and *Sellick* [3.62].
- Forward masking data that shows a linear return to baseline after up to 0.2 s [3.63]. There may be compelling evidence that OHCs are the source of forward masking.

We shall discuss each of these, but two related measures are the most important for understanding these NL masking effects, the upward spread of masking (USM) and two-tone suppression (2TS).

Basilar Membrane Nonlinearity. The most basic early and informative of these nonlinear effects was the NL basilar membrane measurements made by *Rhode* [3.46, 47], as shown in Fig. 3.4a, showing that the basilar membrane displacement to be a highly NL function of level. For every four dB of pressure level increase on the input, the output displacement (or velocity) only changed one dB. This compressive nonlinearity depends on frequency, and only occurs near the most sensitive region (e.g., the tip of the tuning curve). For other frequencies the system was either linear, namely, one dB of input change gave one dB of output change for frequencies away from the best frequency, or very close to linear. This NL effect was highly dependent on the health of the animal, and would decrease or would not be present at all, when the animal was not in its physiologically pristine state.

An important and useful measure of cochlear linear and nonlinear response first proposed by *Rhode* [3.46, Fig. 8], is shown in Fig. 3.4b which describes cochlear tuning curves by straight lines on log–log coordinates. Such straight line approximations are called *Bode plots* in the engineering literature. The *slopes* and *break points*, defined as the locations where the straight lines cross, characterize the response.

Otoacoustic Emissions. A few years after Rhode's demonstration of cochlear nonlinearity, David Kemp observed otoacoustic emissions (tonal sound emanating from the cochlea and NL *echos* to clicks and tone bursts) [3.49, 54, 64–66]. Kemp's findings were like a jolt to the field, which led to a cottage industry of objective testing of the auditory system, including both cochlear and middle ear tests.

Motile OHCs. Subsequently, *Brownell* et al. [3.50] discovered that isolated OHCs change their length when placed in an electric field, thus that the outer hair cell is motile. This then led to the intuitive and widespread proposal that outer hair cells act as voltage-controlled motors that directly drive the basilar membrane on a cycle by cycle basis. It seems quite clear, from a great deal of data, that the OHC onset response time is on the order of one cycle or so of the BM impulse response, because the first peak is linear [3.67]. The release time must be determined by the OHC membrane properties, which is slow relative to the attack. Thus OHC NL processing is the basis for both the frequency asymmetry of simultaneous (upward versus downward spread) and temporal (forward versus backward) masking.

As summarized in Fig. 3.5, OHCs provide feedback to the BM via the OHC receptor potential, which in turn is modulated by both the position of the basilar

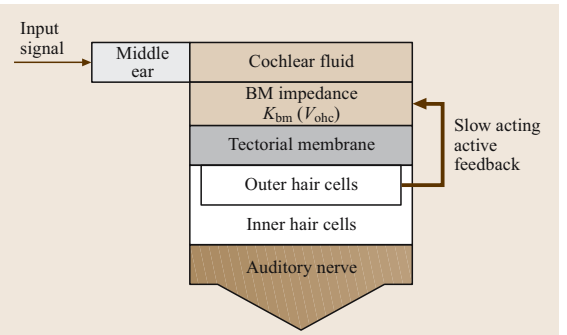


Fig. 3.5 Block flow diagram of the inner ear (after *Allen* [3.51])

membrane (forming a fast feedback loop), and alternatively by the efferent neurons that are connected to the outer hair cells (forming a slow feedback loop). The details of all this are the topic of a great deal of present research.

OHCs are the one common element that link all the NL data previously observed, and a missing piece of the puzzle that most needs to be understood before any model can hope to succeed in predicting basilar membrane, hair cell, and neural tuning, or NL compression. Understanding the outer hair cell's two-way mechanical transduction is viewed as the key to solving the problem of the cochlea's dynamic range.

Historically the implication that hair cells might play an important role in cochlear mechanics go back at least to 1936 when loudness recruitment was first reported by *Fowler* [3.68] in a comment by *R. Lorente de Nó* [3.14] stating that cochlear hair cells are likely to be involved in loudness recruitment.

The same year *Steinberg* and *Gardner* [3.3] were explicit about the action of recruitment when they concluded:

> *When someone shouts, such a deafened person suffers practically as much discomfort as a normal hearing person would under the same circumstances. Furthermore for such a case, the effective gain in loudness afforded by amplification depends on the amount of variable type loss present. Owing to the expanding action of this type of loss it would be necessary to introduce a corresponding compression in the amplifier in order to produce the same amplification at all levels.*

Therefore as early as 1937 there was a clear sense that cochlear hair cells were related to dynamic range compression.

More recently, theoretical attempts to explain the difference in tuning between normal and damaged cochleae led to the suggestion that OHCs could influence BM mechanics. In 1983 *Neely* and *Kim* [3.69] concluded:

> *We suggest that the negative damping components in the model may represent the physical action of outer hair cells, functioning in the electrochemical environment of the normal cochlea and serving to boost the sensitivity of the cochlea at low levels of excitation.*

In 1999 yet another (a fourth) important discovery was made, that the outer-hair-cell mechanical stiffness depends on the voltage across its membrane [3.70, 71]. This change in stiffness, coupled with the naturally occurring internal static pressure, may well account for the voltage dependent accompanying length changes (the cell's voltage dependent motility). This view follows from the block diagram feedback model of the organ of Corti shown in Fig. 3.5 where the excitation to the OHC changes the cell voltage V_{ohc}, which in turn changes the basilar stiffness [3.51]. This is one of several possible theories that have been put forth.

This experimental period set the stage for explaining the two most dramatic NL measures of cochlear response, the upward spread of masking and its related neural correlate, two-tone suppression, and may well turn out to be the explanation of the nonlinear forward-masking effect as well [3.63].

Simultaneous Dynamic-Masking

The psychophysically measured *upward spread of masking* (USM) and the neurally measured *two-tone suppression* (2TS) are closely related dynamic-masking phenomena. Historically these two measures have been treated independently in the literature. As will be shown, it is now clear that they are alternative objective measures of the same OHC compressive nonlinearity. Both involve the dynamic suppression of a basal (high-frequency) probe due to the simultaneous presentation of an apical (low-frequency) suppressor. These two views (USM versus 2TS) nicely complement each other, providing a symbiotic view of cochlear nonlinearity.

Upward Spread of Masking (USM). In a classic paper, *Mayer* [3.72] was the first to describe the asymmetric nature of masking [3.63, 73]. Mayer made his qualitative observations with the use of clocks, organ pipes and tuning forks, and found that that the spread of masking is a strong function of the probe-to-masker frequency ratio (f_p/f_m) [3.63].

In 1923, Fletcher published the first quantitative results of tonal masking. In 1924, Wegel and Lane extended Fletcher's experiments (Fletcher was the subject [3.27, p. 325]) using a wider range of tones. Wegel and Lane then discuss the results in terms of their 1-D model described above. As shown in Fig. 3.6a, Wegel and Lane's experiments involved presenting listeners with a masker tone at frequency $f_m = 400\,\text{Hz}$ and intensity I_m (the abscissa), along with a probe tone at frequency f_p (the parameter used in the figure). At each masker intensity and probe frequency, the threshold probe intensity $I_p^*(I_m)$ is determined, and displayed relative to its threshold *sensation level* (SL) (the ordinate

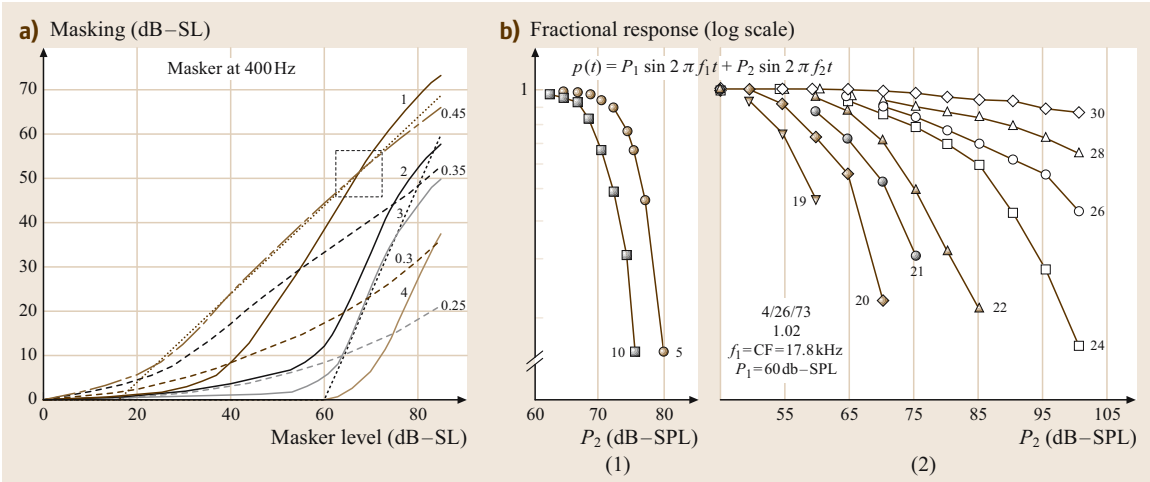


Fig. 3.6a,b On the left **(a)** we see the psychoacoustic measure of 2TS, called the upward spread of masking. On the right **(b)** are related measures taken in the auditory nerve by a procedure called two-tone suppression (2TS). Low- and high-side masking or suppression have very different thresholds and slopes. These suppression slopes and thresholds are very similar between 2TS and the USM. **(a)** Upward spread of masking as characterized by Wegel and Lane in 1924. The *solid lines* correspond to the probe being higher than the 400 Hz masker, while the *dashed lines* correspond to the 400 Hz probe lower than the masker. On the *left* we see upward spread of masking functions from Wegel and Lane for a 400 Hz low-frequency masker. The abscissa is the masker intensity I_m in dB − SL while the ordinate is the threshold probe intensity $I_p^*(I_m)$ in dB − SL. The frequency of the probe f_p, expressed in kHz, is the parameter indicated on each curve. The *dashed box* shows that the masking due to a 1 kHz tone becomes more than that at 450 Hz, for a 400 Hz probe. This is the first observation of *excitation pattern migration* with input intensity. **(b)** Two-tone suppression (2TS) input–output (IO) functions from *Abbas* and *Sachs* [3.60, Fig. 8]. On the *left* (1) is low-side suppression and on the *right* (2) we see high-side suppression. In 2TS the suppressor plays the role of the masker and the probe the role of the maskee. Note that the threshold of suppression for low-side suppressor (masker) is close to 70 dB − SPL, which is similar to human low-side suppressors, the case of the Wegel and Lane USM (1) (60–70 dB − SPL). The onset of suppression for high-side suppressors is close to the neuron's CF threshold of 50 dB, as elaborated further in Fig. 3.7a

is the probe level at threshold [dB − SL]). The asterisk indicates a threshold measure.

In Fig. 3.6a $f_m = 400$ Hz, I_m is the abscissa, f_p is the parameter on each curve, in kHz, and the threshold probe intensity $I_p^*(I_m)$ is the ordinate. The dotted line superimposed on the 3 kHz curve $(I_m/10^{60/10})^{2.4}$ represents the suppression threshold at 60 dB − SL which has a slope of 2.4 dB/dB. The dotted line superimposed on the 0.45 kHz curve has a slope of 1 and a threshold of 16 dB − SL.

Three regions are clearly evident: the *downward spread* of masking ($f_p < f_m$, dashed curves), *critical band* masking ($f_p \approx f_m$, dashed curve marked 0.45), and the *upward spread* of masking ($f_p > f_m$, solid curves) [3.74].

Critical band masking has a slope close to 1 dB/dB (the superimposed dotted line has a slope of 1). Four years later *Riesz* [3.75] shows critical band masking obeys the *near miss to Weber's law*, as described in Sect. 3.3.2. The downward spread of masking (the dashed lines in Fig. 3.6a) has a low threshold intensity and a variable slope that is less than one dB/dB, and approaches 1 at high masker intensities. The upward spread of masking (USM), shown by the solid curves, has a threshold near 50 dB re sensation level (e.g., 65 dB − SPL), and a growth just less than 2.5 dB/dB. The dotted line superimposed on the $f_p = 3$ kHz curve has a slope of 2.4 dB/dB and a threshold of 60 dB − SL.

The dashed box shows that the upward spread of masking of a probe at 1 kHz can be greater than the masking within a critical band (i. e., $f_p = 450$ Hz $> f_m = 400$ Hz). As the masker frequency is increased, this *crossover effect* occurs in a small frequency region (i. e., 1/2 octave) above the masker frequency. The crossover is a result of a well-documented NL *response migration*, of the excitation pattern with

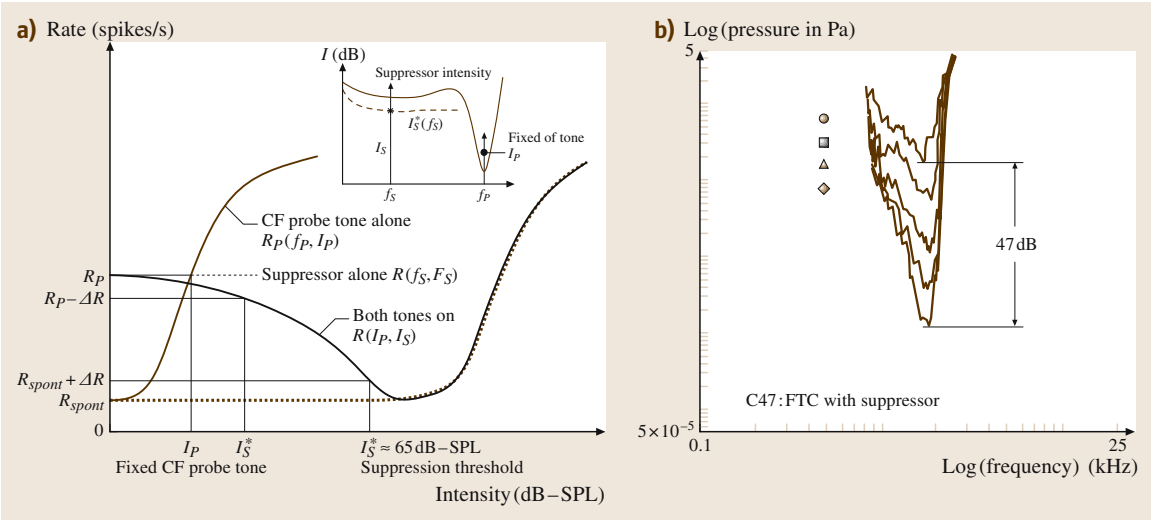


Fig. 3.7 **(a)** Definitions of 2TS low-side masking procedure (see (3.13) and (3.14)). **(b)** Example of 2TS (low-side masking) in the cat auditory nerve (AN). A cat neural tuning curve taken with various *low-side* suppressors present (suppressor below the best frequency), as indicated by the symbols. The tuning curve with the lowest threshold is for no suppressor. When the suppressor changes by 20 dB, the F_{cf} threshold changes by 36 dB. Thus for a 2 kHz neuron, the slope is 36/20, or 1.8. These numbers are similar to those measure by *Delgutte* [3.80]. One Pa = 94 dB − SPL

stimulus intensity, described in a wonderful paper by *McFadden* [3.76]. Response migration was also observed by *Munson* and *Gardner* in a classic paper on forward masking [3.77]. This important migration effect is beyond the scope of the present discussion, but is reviewed in [3.74, 78, 79] discussed in the caption of Fig. 3.10.

The upward spread of masking is important because it is easily measured psychophysically in normal hearing people, is robust, well documented, and nicely characterizes normal outer-hair-cell nonlinearities. The psychophysically measured USM has correlates in basilar membrane and hair cell signals, and is known as two-tone suppression (2TS) in the auditory nerve literature, as discussed in the caption of Fig. 3.6b.

Two-Tone Suppression. The neural correlate of the psychophysically measured USM is called *two-tone suppression* (2TS). As shown in the insert of Fig. 3.7a, first a neural tuning curve is measured. A pure tone probe at intensity $I_p(f_p)$, and frequency f_p, is placed a few dB (e.g., 6 to 10) above threshold at the characteristic (best) frequency of the neuron F_{cf} (i. e., $f_p = F_{cf}$). In 2TS a suppressor tone plays the role of the masker. There are two possible *thresholds*. The intensity of the suppressor tone $I_s(f_s)$ at frequency f_s is increased until either

1. the rate response to either the probe alone $R(I_p, I_s = 0)$ decreases by a small increment Δ_R, or
2. drops to the small increment Δ_R, just above the undriven spontaneous rate $R(0, 0)$.

These two criteria are defined in Fig. 3.6b and may be written

$$R_p\left(I_p, I_s^*\right) \equiv R(I_p, 0) - \Delta_R \tag{3.13}$$

and

$$R_{spont}\left(I_p, I_s^*\right) \equiv R(0, 0) + \Delta_R\,; \tag{3.14}$$

Δ_R indicates a fixed small but statistically significant constant change in the rate (e.g., $\Delta_R = 20$ spikes/s is a typical value). The threshold suppressor intensity is defined as $I_s^*(f_s)$, and as before the $*$ indicates the threshold suppressor intensity. The two threshold definitions (3.13) and (3.14) are very different, and both are useful. The difference in intensity between the two thresholds is quite large, and the more common measure used by *Abbas* and *Sachs* [3.60] is (3.13). The second measure (3.14) is consistent with neural tuning curve suppression, and is therefore the more interesting of the two. It corresponds to suppression of the probe to threshold.

Neural data of *Abbas* and *Sachs* [3.60, Fig. 8] are reproduced in Fig. 3.6b. For this example (see entry in lower-right just below 105), F_{cf} is 17.8 kHz, and the

$f_p = F_{cf}$ probe intensity $20 \log 10(|P_1|)$ is 60 dB. The label on the curves is the frequency f_1. The threshold intensity of the associated neural tuning curve is has a low spontaneous rate and a 50–55 dB threshold. The left panel of Fig. 3.6b is for apical suppressors that are lower in frequency than the characteristic frequency (CF) probe ($f_s < f_p$). In this case the threshold is just above 65 dB − SPL. The suppression effect is relatively strong and almost independent of frequency. In this example the threshold of the effect is less than 4 dB apart (the maximum shift of the two curves) at suppressor frequencies f_s of 10 and 5 kHz (a one octave separation).

The right panel shows the case $f_s > f_p$. The suppression threshold is close to the neuron's threshold (e.g., 50 dB − SPL) for probes at 19 kHz, but increases rapidly with frequency. The strength of the suppression is weak in comparison to the case of the left panel ($f_s < f_p$), as indicated by the slopes of the family of curves.

The Importance of the Criterion. The data of Fig. 3.6b uses the first suppression threshold definition (3.13) R_p (a small drop from the probe driven rate). In this case the F_{cf} probe is well above its detection threshold at the suppression threshold, since according to definition (3.13), the probe is just detectably reduced, and thus audible. With the second suppression threshold definition (3.14), the suppression threshold corresponds to the detection threshold of the probe. Thus (3.14), *suppression to the spontaneous rate*, is appropriate for Wegel and Lane's masking data where the probe is at its detection threshold $I_p^*(I_m)$. Suppression threshold definition (3.14) was used when taking the 2TS data of Fig. 3.7b, where the suppression threshold was estimated as a function of suppressor frequency.

To be consistent with a detection threshold criterion, such as the detection criterion used by Wegel and Lane in psychophysical masking, (3.14) must be used. To have a tuning curve pass through the F_{cf} probe intensity of a 2TS experiment (i. e., be at threshold levels), it is necessary to use the suppression to rate criterion given by (3.14). This is shown in Fig. 3.7b where a family of tuning curves is taken with different suppressors present. As described by *Fahey* and *Allen* [3.56, Fig. 13], when a probe is placed on a specific tuning curve of Fig. 3.7b, corresponding to one of the suppressor level symbols of Fig. 3.7b, and a suppression threshold is measured, that suppression curve will fall on the corresponding suppression symbol of Fig. 3.7b. There is a symmetry between the tuning curve measured in the presents of a suppressor, and a suppression threshold obtained with a given probe. This symmetry only holds for criterion (3.14), the detection threshold criterion, which is appropriate for Wegel and Lane's data. If one uses (3.13) as in [3.60] they will not see this symmetry as cleary.

Suppression Threshold. Using the criterion (3.14), *Fahey* and *Allen* [3.56, Fig. 13] showed that the suppression threshold $I_s^*(I_p)$ in the tails is near 65 dB − SPL (0.04 Pa). This is true for suppressors between 0.6 and 4 kHz. A small amount of data are consistent with the threshold being constant to much higher frequencies, but the Fahey and Allen data are insufficient on that point.

Suppression Slope. *Delgutte* has written several insightful papers on masking and suppression [3.80–82]. He estimated how the intensity growth slope (in dB/dB) of 2TS varies with suppressor frequency for several probe frequencies [3.80]. As may be seen in his figure, the suppression growth slope for the case of a low frequency apical suppressor on a high frequency basal neuron (the case of the left panel of Fig. 3.6b), is ≈ 2.4 dB/dB. This is the same slope as for Wegel and Lane's 400 Hz masker, 3 kHz probe USM data shown in Fig. 3.6a. For suppressor frequencies greater than the probe's ($f_s > f_p$), Delgutte reports a slope that is significantly less than 1 dB/dB. Likewise Wegel and Lane's data has slopes much less than 1 for the downward spread of masking.

One may conclude that USM and 2TS data show systematic and quantitative correlations between the threshold levels and slopes. The significance of these correlations has special importance because

1. they come from very different measurement methods, and
2. Wegel and Lane's USM are from human, while the 2TS data are from cat, yet they show similar responses. This implies that the cat and human cochleae may be quite similar in their NL responses.

The USM and 2TS threshold and growth slope (e.g., 50 dB − SL and 2.4 dB/dB) are important features that must be fully understood and modeled before we can claim to understand cochlear function. While there have been several models of 2TS [3.83–85] as discussed in some detail by *Delgutte* [3.80], none are in quantitative agreement with the data. The two-tone suppression model of *Hall* [3.84] is an interesting contribution to this problem because it qualitatively explores many of the key issues. Finally forward-masking data also show related nonlinear properties that we speculate may turn out to be related to NL OHC function as well [3.78, 86, 87].

3.2.2 Outer-Hair-Cell Transduction

The purpose of this section is to address two intimately intertwined problems *cochlear frequency selectivity* and *cochlear nonlinearity*. The fundamental question in cochlear research today is: *What is the role of the outer hair cell (OHC) in cochlear mechanics?* The OHC is the source of the NL effect, and the end product is dynamic masking, including the USM, 2TS and forward masking, all of which include dramatic amounts of gain and tuning variation. The issues are the nature of the NL transformations of the BM, OHC cilia motion, and OHC soma motility, at a given location along the basilar membrane.

The prevailing and popular *cochlear-amplifier* view is that the OHC provides *cochlear sensitivity* and *frequency selectivity* [3.5, 88–94]. The alternative view, argued here, is that the OHC compresses the excitation to the inner hair cell, thereby providing dynamic range expansion.

There is an important difference between these two views. The *first* view deemphasizes the role of the OHC in providing dynamic range control (the OHC's role is to improve sensitivity and selectivity), and assumes that the NL effects result from OHC saturation.

The *second* view places the dynamic range problem as the top priority. It assumes that the sole purpose of the OHC nonlinearity is to provide dynamic range compression, and that the OHC plays no role in either sensitivity or selectivity, which are treated as important but independent issues. Of course other views besides these two are possible.

The Dynamic-Range Problem

The question of how the large (up to 120 dB) dynamic range of the auditory system is attained has been a long standing problem which remains fundamentally incomplete. For example, *recruitment*, the most common symptom of neurosensory hearing loss, is best characterized as the loss of dynamic range [3.3, 10, 15, 95]. Recruitment results from outer-hair-cell damage [3.96]. To successfully design hearing aids that deal with the problem of recruitment, we need models that improve our understanding of *how* the cochlea achieves its dynamic range.

Based on a simple analysis of the IHC voltage, one may prove that the dynamic range of the IHC must be less than 65 dB [3.97]. In fact it is widely accepted that IHC dynamic range is less than 50 dB.

The IHC's transmembrane voltage is limited at the high end by the cell's open circuit (unloaded) membrane voltage, and at the low end by thermal noise. There are two obvious sources of thermal noise, cilia Brownian motion, and Johnson (shot) noise across the cell membrane (Fig. 3.8).

The obvious question arises: *How can the basic cochlear detectors (the IHCs) have a dynamic range of less than* 50 dB *(a factor of* 0.3×10^2*), and yet the auditory system has a dynamic range of up to* 120 dB

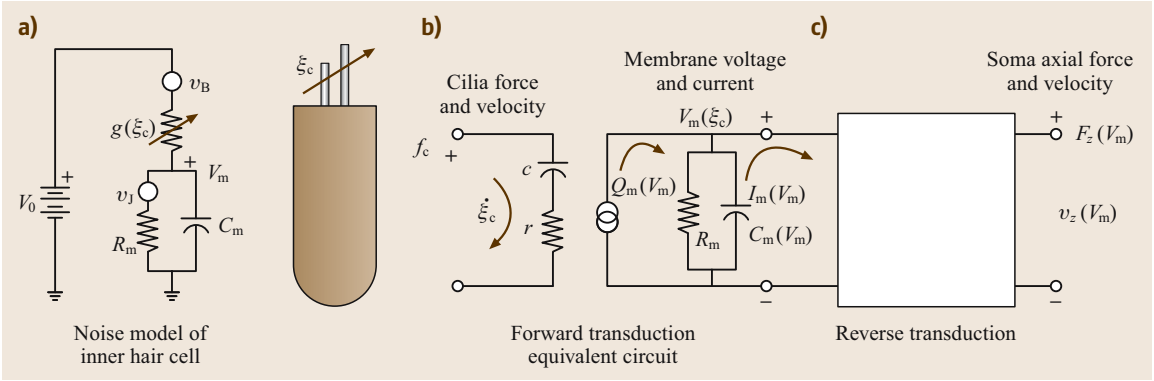


Fig. 3.8a–c On the far left **(a)** is the electrical equivalent circuit model of an IHC with thermal noise sources due to the cell leakage resistance Johnson and shot noise v_J and the Brownian motion of the cilia, represented by the voltage noise source v_B. The cilia force f_c and velocity $\dot{\xi}_c$ are the stimulus (input) variables to the forward transduction **(b)**, and are loaded by the mechanical impedance of the cilia viscous drag r and compliance c. **(c)** For OHCs, when the cilia move, current flows into the cell charging the membrane capacitance, thus changing the membrane voltage V_m. This membrane capacitance $C_m(V_m)$ is voltage dependent (i. e., it is NL). The membrane voltage has also been shown to control the cell's soma axial stiffness. It follows that the axial force $F_z(V_m)$ the cell can deliver, and the axial velocity $\mathcal{V}_z(V_m)$ of the cell, must also depend on the membrane voltage. The precise details of how all this works is unknown

(a factor of 10^6*)?* The huge amount of indirect evidence has shown that this increased dynamic range results from mechanical NL signal compression provided by outer hair cells. This dynamic-range compression shows up in auditory psychophysics and in cochlear physiology in many ways.

This thus forms the basic dynamic-range dilemma.

Outer-Hair-Cell Motility Model

A most significant finding in 1985 was of OHC *motility*, namely that the OHC changes its length by up to 5% in response to the cell's membrane voltage [3.50, 99, 100]. This less than 5% change in length must account for a 40 dB (100 times) change in cochlear sensitivity. This observation led to a significant increases in research on the OHC cell's motor properties.

In 1999 it was shown that the cell's longitudinal soma stiffness changes by at least a factor of 2 ($> 100\%$), again as a function of cell membrane voltage [3.70, 71]. A displacement of the cilia in the direction of the tallest cilia, which is called a *depolarizing* stimulus, decreases the magnitude of the membrane voltage $|V_m|$, *decreases* the longitudinal soma stiffness, and *decreases* the cell soma length. A hyperpolarizing stimulus increases the stiffness and extends the longitudinal soma length.

Given this much larger relative change in stiffness (a factor of 2) compared to the relative change in length (a factor of 1.05), for a maximum voltage change, it seems possible, or even likely, that the observed length changes (the motility) are simply a result of the voltage dependent stiffness. For example, imagine a spring stretched by applying a constant force (say a weight), and then suppose that the spring's stiffness decreases. It follows from Hooke's law (3.5) that the spring's length will *increase* when the stiffness decreases.

Each cell is stretched by its internal static pressure $\mathcal{P}$ [3.101], and its stiffness is voltage controlled [3.70, 71]. The voltage dependent relative stiffness change is much greater than the relative motility change. Thus we have the necessary conditions for stiffness-induced motility.

3.2.3 Micromechanics

Unlike the case of macromechanical models, the physics of every micromechanical model differs significantly. This is in part due to the lack of direct experimental evidence of physical parameters of the cochlea. This is an important and very active area of research (e.g., [3.102]).

To organize our discussion of cochlear micromechanics, we represent each radial cross-section through the cochlear partition (Fig. 3.1b) as a linear two-port network. A general formalization in transmission matrix form of the relation between the basilar membrane *input* pressure $P(x, s)$ and velocity $V(x, s)$ and the OHC *output* cilia bundle shear force $f(x, s)$ and shear velocity

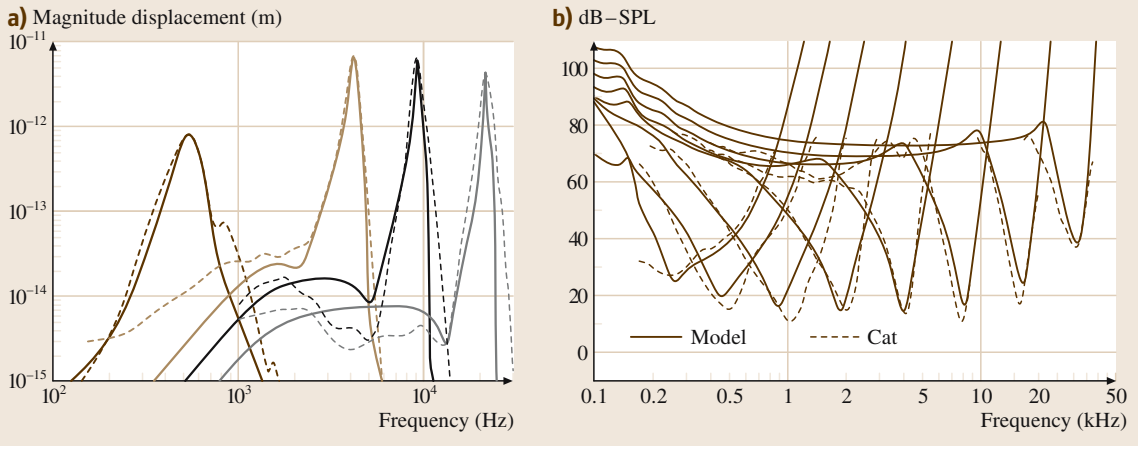


Fig. 3.9a,b The tuning curves shown by the dashed lines are the average of single nerve fiber responses from six cats obtained by M. C. Liberman and B. Delgutte. **(a)** Comparison between neural data and the computed model excitation patterns from Allen's passive RTM model (transfer function format). This CA model assumes an IHC cilia bundle displacement of about 50 pm at the neural rate threshold. **(b)** Comparison between neural data computed tuning curves from *Neely*'s active model [3.98]. This CA model assumes an IHC cilia bundle displacement of 300 pm (0.3 nm) at the neural rate threshold

$v(x, s)$

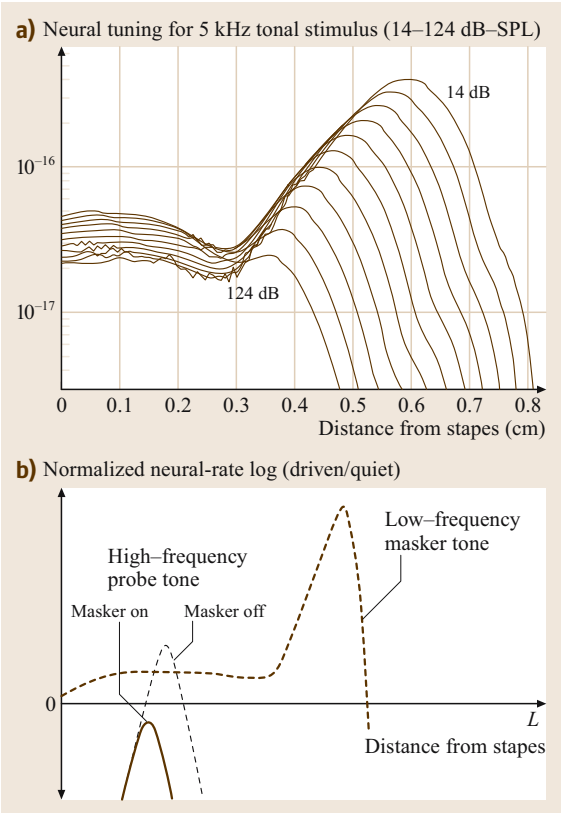
$$\begin{pmatrix} P \\ V \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} f \\ v \end{pmatrix} , \qquad (3.15)$$

where A, B, C, and D are complex functions of place X and radian frequency s.

Passive BM Models

The most successful *passive* model of cochlear tuning is the resonant tectorial membrane (RTM) model [3.9, 104]. The RTM model starts from the assumption that the slope S_2 of BM tuning is insufficient to account for the slope S_2 of neural tuning, as seen in Fig. 3.4b. This sharpening is accounted for by a reflection in the tectorial membrane, introducing an antiresonance (*spectral zero*) at frequency F_z (Fig. 3.4b), which is about half an octave below the resonant frequency F_{cf} of the basilar membrane. As described by *Allen* and *Neely* [3.9], the detailed A, B, C, D elements of (3.15) are given by *Allen* [3.104], *Allen* and *Neely* [3.9].

As described in *Allen* [3.105], the response ratio of IHC cilia bundle displacement to basilar membrane displacement is defined as $H_{ihc}(x, s)$. The parameters of the RTM model may be chosen such that model results fit the experimental neural threshold tuning curves closely, as shown in Fig. 3.9a.



The Nonlinear RTM Model. The resonant tectorial membrane (RTM) model is made NL by control of the BM stiffness via OHC's stiffness is based on Fig. 3.1b. The OHC soma stiffness has been shown to be voltage dependent by *Dallos* et al. [3.106] and dependent on prestin in the membrane wall [3.107]. If an elastic connection is assumed where the TM attaches to the Limbus, and if this elasticity is similar to that of the cilia of the OHC, then the resulting transfer function between the BM and IHC cilia is strongly filtered at low frequencies [3.51,103,108,109]. Such models are actively under consideration [3.102].

It is postulated that the decrease in OHC stiffness accompanying cilia stimulation results in a decrease of the net BM partition stiffness $K_p(x)$ (i. e., increasing compliance) of (3.6). As shown in Fig. 3.3, this decrease in the local BM stiffness would result in the partition excitation pattern shifting basally towards the stapes. Such shifts in the BM response patterns are commonly seen. Another way to view this is shown in Fig. 3.10. This migration of the excitation pattern, combined with the assumption that the TM has a high-pass characteristic, means that the cilia excitation gain at CF is nonlinearly compressed

Fig. 3.10a,b In **(a)** results of model calculations by *Sen* and *Allen* [3.103] are shown of a NL BM stiffness model. On the right shows a cartoon of what might happen to the excitation pattern of a low-level probe when a suppressor is turned on given such a nonlinearity. The presence of the suppressor causes the probe to be suppressed and shifted slightly toward the base when the stiffness is decreased with increased level. It may be inferred from Fig. 3.3a that, if the BM stiffness is reduced, the location of the maximum will shift to the base, as is seen in real data. **(a)** Compression in the NL RTM model. Note how the response at the peak is reduced as the BM stiffness changes, causing the peak to shift to the base. As this happens the response in the tail region between $0 \le X \le 0.3$ cm becomes more sensitive, and thus shows an expansive NL response. All of these effects have been seen in real BM data. **(b)** Cartoon showing the effect of a low-side masker on a high-frequency tone as a function of position along the basilar membrane. When the suppressor is turned on, the CF of the high-frequency probe becomes less sensitive and shifts to higher frequencies. We model this effect in the panel on the left as BM stiffness that depends on level (i. e., $K_p(I_s)$)

as the intensity increases. This compression effect is shown in a cartoon format in Fig. 3.10b, while Fig. 3.10a shows the actual calculated model results. Note how the bandwidth $\Delta_f(X)$) remains approximately constant as a function of input intensity.

Sewell [3.110] has nicely demonstrated that as the voltage driving the hair cells changes, the neural gain in dB at CF changes proportionally. It is not yet known why the dB gain is proportional to the voltage (1 dB/mv), however this would explain why forward masking decays linearly in dB value with time, after a strong excitation, since the membrane voltage $V_m(t)$ is proportional to e^{-t/τ_m}, due to the OHC membrane's $\tau_m = RC$ time constant. In my view, explaining the proportionality between the neural threshold in dB and the linear membrane voltage, is key.

Discussion. Two important advantages of the NL RTM model include its physically based assumptions (described above), and its simplicity. Given these physical assumptions, we show next that the NL RTM model can explain:

1. the basal-ward half-octave traveling-wave migration as a function of increasing intensity [3.76],
2. the upward spread of masking (USM) [3.20, 21], two-tone suppression (2TS) (see Sect. 3.2.1),
3. distortion product generation [3.49,55,56,111–113],
4. normal and recruiting loudness growth, and
5. hypersensitive tails [3.45].

From the steep 2.5 dB/dB slope of the USM and 2TS (Fig. 3.6a) it seems necessary that the low-frequency suppressor is turning down the high-frequency probe even though the growth of the masker at the high frequency's place is linear with masker level, as shown in Fig. 3.10b.

Active BM Models

One obvious question about active cochlear models is *Are they really necessary?* At least three attempts to answer this question based on detailed comparisons of basilar membrane responses have concluded that the measured responses *cannot* be accounted for by a passive cochlear model [3.93, 114–117].

The CA Hypothesis. The most popular active micromechanical theory is called the *cochlear amplifier* (CA) hypothesis. The concept of the *cochlear amplifier*, originated by Gold, Kemp, Kim and Neely, and named by H. Davis, refers to a hypothetical mechanism within the cochlear partition which increases the *sensitivity* of basilar membrane vibrations to low-level sounds and, at the same time, increases the *frequency selectivity* of these vibrations [3.94]. The CA adds mechanical energy to the cochlear partition at acoustic frequencies by drawing upon the electrical and mechanical energy available from the outer hair cells. In response to a tone, the CA adds mechanical energy to the cochlear traveling wave in the region defined by S_2 as it approaches the place of maximum response. This energy is reabsorbed at other places along the cochlear partition. The resulting improvement in sensitivity of the ear due to the CA is thought to be 40 dB, or more under certain conditions; however, the details of how this amplification might be accomplished are still unknown [3.118, 119]. A general discussion of this model is presented in *Geisler* [3.90], and in *Allen* and *Fahey* [3.91].

It is presumed that this OHC action amplifies the BM signal energy on a cycle-by-cycle basis, increasing the sensitivity [3.69, 92]. In some of the models it is assumed that this cycle-by-cycle pressure (force) due to the OHCs causes the sharp BM tuning tip. In most of these models, the CA is equivalent to introducing a frequency-dependent negative damping (resistance) into the BM impedance [3.120]. Nonlinear compression is introduced by assuming that the resistance is signal level dependent. This NL resistance model was first described by *Hall* [3.84] for the case of $R > 0$. Thus the CA model is an extension of Hall's model to the case of $R < 0$. In several models NL negative damping is obtained with a nonlinear stiffness and a small delay. The addition of a small delay introduces a negative real part into the impedance. In mathematical physics, NL damping resonators are described by *van der Pol equations*, while NL stiffness resonators are described by *Duffing equations* [3.121].

Allen and *Fahey* [3.91] developed a method for directly measuring the cochlear amplifier (CA) gain. All of the studies to date using this method have found no gain. However many researchers continue to believe that the CA has gain. Given that the gain is order 40–50 dB this is difficult to understand. A nice summary of this situation has been recently published in *Shera* and *Guinan* [3.120]. The reasons for the failure to directly measure any CA gain are complex and multifaceted, and many important questions remain open. One possibility that remains open is that the many observed large NL OHC BM effects we see are not due to cycle-by-cycle power amplification of the BM traveling wave.

Discussion and Summary

Discussion. Both active and passive BM models are reasonably successful at simulating the neural thresh-

old response tuning curves. Thus we may need to look elsewhere to contrast the difference between these two approaches, such as 2TS/USM. While the passive RTM model is easily made NL with the introduction of $K_{ohc}(V_m)$, differences between *nonlinear* RTM and CA models have not yet been investigated. The CA and RTM models differ in their interpretation of damaged cochlear responses. In CA models, the loss of sensitivity of the cochlea with damage is interpreted as a loss of CA gain while in passive models, the loss of sensitivity has been interpreted as a 2:1 change in the BM stiffness [3.122].

The discovery of OHC motility demonstrates the existence of a potential source of mechanical energy within the cochlear partition which is suitably positioned to influence vibrations of the basilar membrane. It is still an open question whether this source of energy is sufficient to power a CA at high frequencies.

One possible advantage of the CA is that of improving the signal-to-noise ratio in front of the IHC detector. A weakness of the CA models has been their lack of specificity about the physical realization of the active elements. Until we have a detailed physical representation for the CA, RTM models have the advantage of being simpler and more explicit.

The discovery by He and Dallos that the OHC soma stiffness is voltage dependent is an exciting development for the NL passive RTM model, as it greatly simplifies the implementation of the physical model. The RTM model has been in disfavor because many feel it does not account for basilar membrane tuning. This criticism is largely due to the experimental results of physiologists who have measured the BM–ear canal transfer function, and found the tuning of BM velocity to be similar to neural threshold response data. Much of the experimental BM data, however, are not convincing on this point, with the BM slope S_2 (Fig. 3.4b) generally being much smaller than that of neural responses [3.97]. The question of whether an active model is required to simulate measured BM responses is still being debated.

Better estimates of the amplitude of cilia bundle displacement at a given sound pressure level directly address the sensitivity questions. If the estimate of *Russell* of 30 mV/degree is correct [3.123], then the cochlear sensitivity question may be resolved by having very sensitive detectors. Also, better estimates are needed of the ratio of the BM frequency response to the IHC frequency response, both at high and low frequencies. Rhode's approach of using the slopes of Fig. 3.4b rather than traditional ad hoc bandwidth measures, is a useful tool in this regard. The bandwidth 10 dB down relative to the peak has been popular, but arbitrary and thus poor, criterion in cochlear research. A second, somewhat better, bandwidth measure is Fletcher's *equivalent rectangular bandwidth* discussed in *Allen* [3.10].

Summary. This section has reviewed what we know about the cochlea. The *Basics* section reviews the nature of modeling and briefly describes the anatomy of the inner ear, and the function of inner and outer hair cells. In Sect. 3.1.2 we reviewed the history of cochlear modeling. The Wegel and Lane paper was a key paper that introduced the first detailed view of masking, and in the same paper introduced the first modern cochlear model Fig. 3.2b. We presented the basic tools of cochlear modeling, *impedance*, and introduced the *transmission matrix* method (two-port analysis). We describe how these models work in intuitive terms, including how the basilar membrane may be treated as having a frequency dependent acoustic hole. The location of the hole, as a function of frequency, is called the cochlear map. This hole keeps fluid from flowing beyond a certain point, producing the cochlear traveling wave.

We reviewed and summarized the NL measures of cochlear response. Since these data are not fully understood, and have not been adequately modeled, this is the most difficult section. However it is worth the effort to understand these extensive data and to appreciate the various relations between them, such as the close parallel between two-tone suppression and the upward spread of masking, and between loudness recruitment and outer hair cell damage.

We review several models of the hair cell, including forward and reverse transduction. Some of this material is recently published, and the view of these models could easily change over the next few years as we better understand reverse transduction.

Finally in Sect. 3.2.3 we reviewed the basics of micromechanics. We have presented the two basic types of models, *passive* and *active* models, with a critical review of each.

3.3 Neural Masking

When modeling human psychophysics one must carefully distinguish the external *physical* variables, which we call Φ variables, from the internal *psychophysical* variables, or Ψ variables. It may be helpful to note that

Φ and Ψ sound similar to the initial syllable of the words *phy*sical and *psy*chological, respectively [3.124]. Psychophysical modeling seeks a transformation from the Φ domain to the Ψ domain. The Φ intensity of a sound is easily quantified by direct measurement. The Ψ intensity is the loudness. The idea that loudness could be quantified was first suggested by *Fechner* [3.125] in 1860, who raised the question of the quantitative transformation between the physical and psychophysical intensity. For a recent review of this problem, and a brief summary of its long history, see *Schlauch* et al. [3.126]. This section is based on an earlier report by *Allen* [3.79], and *Allen* and *Neely* [3.127].

An increment in the intensity of a sound that results in a *just noticeable difference* is called an intensity JND. Fechner suggested quantifying the intensity-loudness growth transformation by counting the number of the *loudness JNDs* between two intensity values. However, after many years of work, the details of the relationship between loudness and the intensity JNDs remain unclear [3.128–130].

The contribution of *Allen* and *Neely* [3.127] and *Allen* [3.79] is that it takes a new view of the problem of the intensity JND and loudness by merging the 1953 Fletcher neural excitation pattern model of loudness [3.10, 131] with auditory signal detection theory [3.132].

It is generally accepted that the intensity JND is the physical correlate of the psychological-domain uncertainty corresponding to the psychological intensity representation of a signal. Along these lines, for long duration pure tones and wide-band noise, we assume that the Ψ-domain intensity is the loudness, and that the loudness JND results from loudness *noise* due to its stochastic representation.

To model the intensity JND we must define a *decision variable* associated with loudness and its random fluctuations. We call this loudness random decision variable the *single-trial loudness*. Accordingly we define the loudness and the loudness JND in terms of the first and second moments of the single-trial loudness, that is the mean and variance of the distribution of the single-trial loudness decision variable. We also define the ratio of the mean loudness to the loudness standard deviation as the *loudness signal-to-noise ratio*, $\mathrm{SNR_L}$.

Our ultimate goal in this work is to use signal detection theory to unify masking and the JND, following the 1947 outline of this problem by *Miller* [3.133]. Tonal data follows the *near miss to Weber's law* (thus does not obey Weber's law), while the wide-band noise data does obey Weber's law. We will show that the transformation of the Φ-domain (intensity) JND data (both tone and noise) into the Ψ domain (loudness) unifies these two types of JND data, since $\mathrm{SNR_L}(L)$ is the same for both the tone and noise cases. To help understand these results, we introduce the concept of a near miss to Stevens' law, which we show cancels the near–miss to Weber's law, giving the invariance in $\mathrm{SNR_L}$ for the tone case [3.127]. This work has applications in speech and audio coding.

For the case of tones, we have chosen to illustrate our theoretical work using the classical intensity modulation measurements of *Riesz* [3.75] who measured the intensity JND using small, low-frequency (3 Hz), sinusoidal modulation of tones. *Modern* methods generally use *pulsed* tones which are turned on and off somewhat abruptly, to make them suitable for a two-alternative, forced-choice (2AFC) paradigm. This transient could trigger cochlear forward masking. Riesz's modulation method has a distinct advantage for characterizing the internal signal detection process, because it maintains a nearly steady-state small-signal condition within the auditory system, minimizing any cochlear forward masking component. The interpretation of intensity JNDs is therefore simplified since underlying stochastic processes are stationary.

An outline of this neural masking section is as follows. After some basic definitions in Sect. 3.3.1 and a review of historical models (e.g., Weber and Fechner), in Sect. 3.3.2, we explore issues surrounding the relation between the intensity JND and loudness, for the special cases of tones in quiet and for wide-band noise. First, we look at formulae for counting the number of intensity and loudness JNDs and we use these formulae, together with decision-theoretic principles, to relate loudness to the intensity JND. We then review the loudness-JND theory developed by *Hellman* and *Hellman* [3.134], which provided the inspiration for the present work. Next, we empirically estimate the loudness SNR, defined as the mean loudness over the loudness variance, and proportional to $L/\Delta L$, as a function of both intensity and loudness, using the tonal JND data of *Riesz* [3.75] and the loudness growth function of *Fletcher* and *Munson* [3.41]. We then repeat this calculation for Miller's wide-band noise JND and loudness data. Finally we propose a model of loudness that may be used to compute the JND. This model merges Fletcher's neural excitation pattern model of loudness with signal detection theory.

3.3.1 Basic Definitions

We need a flexible yet clear notation that accounts for important time fluctuations and modulations that are present in the signals, such as beats and gated signals. We include a definition of *masked threshold* because we view the intensity JND as a special case of the masked threshold [3.133]. We include a definition of *beats* so that we can discuss their influence on Riesz's method for the measurement of intensity JNDs.

Intensity. In the time domain, it is common to define the Φ *intensity* in terms of the time-integrated squared signal pressure $s(t)$, namely,

$$I_s(t) \equiv \frac{1}{\varrho c T} \int_{t-T}^{t} s^2(t)\,\mathrm{d}t \,, \tag{3.16}$$

where T is the integration time and ϱc is the specific acoustic impedance of air. The *intensity level* is defined as I_s/I_{ref}, and the *sound pressure level* as $|s|/s_{\mathrm{ref}}$, where the reference intensity is I_{ref} or $10^{-10}\,\mu\mathrm{W/cm^2}$ and the reference pressure $s_{\mathrm{ref}} = 20\,\mu\mathrm{Pa}$. These two reference levels are equivalent at only one temperature, but both seem to be in use. Equivalence of the pressure and intensity references requires that $\varrho c = 40\,\mathrm{cgs}$ Rayls. At standard atmospheric pressure, this is only true when the temperature is about 39 °C. Such levels are typically expressed in dB units.

Intensity of Masker plus Probe. The JND is sometimes called *self-masking*, to reflect the view that it is determined by the internal noise of the auditory system. To model the JND it is useful to define a more-general measure called the *masked threshold*, which is defined in the Φ domain in terms of a nonnegative pressure scale factor α applied to the probe signal $p(t)$ that is then added to the masking pressure signal $m(t)$. The relative intensity of the probe and masker is varied by changing α. Setting $s(t) = m(t) + \alpha p(t)$, we denote the combined intensity as

$$I_{\mathrm{m+p}}(t,\alpha) \equiv \frac{1}{\varrho c T} \int_{t-T}^{t} [m(t) + \alpha p(t)]^2\,\mathrm{d}t \,. \tag{3.17}$$

The unscaled probe signal $p(t)$ is chosen to have the same long-term average intensity as the masker $m(t)$, defined as I. Let $I_{\mathrm{m}}(t)$ be the intensity of the masker with no probe ($\alpha = 0$), and $I_{\mathrm{p}}(t,\alpha) = \alpha^2 I$ be the intensity of the scaled probe signal with no masker. Thus

$$I \equiv I_{\mathrm{m+p}}(t,0) = I_{\mathrm{m}}(t) = I_{\mathrm{p}}(t,1) \,.$$

Because of small fluctuations in I_{m} and I_{p} due to the finite integration time T, this equality cannot be exactly true. We are specifically ignoring these small rapid fluctuations – when these rapid fluctuations are important, our conclusions and model results must be reformulated.

Beats. Rapid fluctuations having frequency components outside the bandwidth of the period T_{second} rectangular integration window are very small and will be ignored (T is assumed to be large). Accordingly we drop the time dependence in terms I_{m} and I_{p}. The beats between $m(t)$ and $p(t)$ of these signals are within a common critical band. Slowly varying correlations, between the probe and masker having frequency components within the bandwidth of the integration window, may *not* be ignored, as with beats between two tones separated in frequency by a few Hz. Accordingly we keep the time dependence in the term $I_{\mathrm{m+p}}(t,\alpha)$ and other slow–beating time dependent terms. In the Φ domain these beats are accounted for as a probe–masker correlation function $\rho_{\mathrm{mt}}(t)$ [3.132, p. 213].

Intensity Increment $\delta I(t,\alpha)$. Expanding (3.17) and solving for the *intensity increment* δI we find

$$\delta I(t,\alpha) \equiv I_{\mathrm{m+p}}(t,\alpha) - I = \left[2\alpha\rho_{\mathrm{mp}}(t) + \alpha^2\right] I \,, \tag{3.18}$$

where

$$\rho_{\mathrm{mp}}(t) = \frac{1}{\varrho c T I} \int_{t-T}^{t} m(t)\,p(t)\,\mathrm{d}t \tag{3.19}$$

defines a normalized cross-correlation function between the masker and the probe. The correlation function must lie between -1 and 1.

Detection Threshold. As the probe-to-masker ratio α is increased from zero, the probe can eventually be detected. We specify the probe *detection threshold* as α_*, where the asterisk indicates the threshold value of α where a subject can discriminate intensity $I_{\mathrm{m+p}}(t,\alpha_*)$ from intensity $I_{\mathrm{m+p}}(t,0)$ 50% of the time, corrected for chance (i. e., obtain a 75% correct score in a direct comparison of the two signals [3.132, p. 129]). The quantity $\alpha_*(t,I)$ is the probe to masker root-mean-square (RMS) pressure ratio at the detection threshold. It is a function of the masker intensity I and, depending on the experimental setup, time. α_* summarizes the experimental measurements.

Masked Threshold Intensity. When $\rho_{mp} = 0$, the *masked threshold intensity* is defined in terms of α_* as

$$I_p^*(I) \equiv I_p(\alpha_*) = \alpha_*^2 I \,,$$

which is the threshold intensity of the probe in the presence of the masker.

The masked threshold intensity is a function of the stimulus modulation parameters. For example, tone maskers and narrow-band noise maskers of equal intensity, and therefore approximately equal loudness, give masked thresholds that are about 20 dB different [3.135]. As a second example, when using the method of beats [3.75], the just–detectable modulation depends on the beat frequency. With *modern* 2AFC methods, the signals are usually gated on and off (100% modulation) [3.136]. According to *Stevens* and *Davis* [3.137, p. 142]

> *A gradual transition, such as the sinusoidal variation used by Riesz, is less easy to detect than an abrupt transition; but, as already suggested, an abrupt transition may involve the production of unwanted transients.*

One must conclude that the *relative masked threshold* [i. e., $\alpha_*(t, I)$] is a function of the modulation conditions, and depends on ρ_{mp}, and therefore T.

Ψ-Domain Temporal Resolution. When modeling time-varying psychological decision variables, the relevant integration time T is not the duration defined by the Φ intensity (3.16), rather the integration time is determined in the Ψ domain. This important Ψ-domain model parameter is called *loudness temporal integration* [3.138]. It was first explicitly modeled by *Munson* in 1947 [3.139].

The Φ-domain temporal resolution (T) is critical to the definition of the JND in Riesz's experiment because it determines the measured intensity of the beats. The Ψ-domain temporal resolution plays a different role. Beats cannot be heard if they are faster than, and therefore *filtered* out by, the Ψ domain response. The Ψ-domain temporal resolution also impacts results for gated stimuli, such as in the 2AFC experiment, though its role is poorly understood in this case. To model the JND as measured by Riesz's method of just-detectable beats, one must know the Ψ-domain resolution duration to calculate the probe–masker effective correlation $\rho_{mp}(t)$ in the Ψ domain. It may be more practical to estimate the Ψ domain resolution from experiments that estimate the degree of correlation, as determined by the beat modulation detection threshold as a function of the beat frequency f_b.

In summary, even though Riesz's modulation detection experiment is technically a masking task, we treat it, following *Riesz* [3.75], *Miller* [3.133], and *Littler* [3.16], as characterizing the intensity JND. It follows that the Ψ-domain temporal resolution plays a key role in intensity JND and masking models.

The Intensity JND ΔI. The intensity *just-noticeable difference* (JND) is

$$\Delta I(I) \equiv \delta(t, \alpha_*) \,, \tag{3.20}$$

the intensity increment at the masked threshold, for the special case where the probe signal is equal to the masking signal ($p(t) = m(t)$). From (3.18) with α set to threshold α_* and $\rho_{mp}(t) = 1$

$$\Delta I(I) = (2\alpha_* + \alpha_*^2) I \,. \tag{3.21}$$

It is traditional to define the intensity JND to be a function of I, rather than a function of $\alpha(I)$, as we have done here. We shall treat both notations as equivalent [i. e., $\Delta I(I)$ or $\Delta I(\alpha)$].

An important alternative definition for the special case of the *pure-tone JND* is to let the masker be a pure tone, and let the probe be a pure tone of a slightly different frequency (e.g., a beat frequency difference of $f_b = 3$ Hz). This was the definition used by *Riesz* [3.75]. Beats are heard at $f_b = 3$ Hz, and assuming the period of 3 Hz is within the passband of the Ψ temporal resolution window, $\rho_{mp}(t) = \sin(2\pi f_b t)$. Thus

$$\Delta I(t, I) = \left[2\alpha_* \sin(2\pi f_b t) + \alpha_*^2\right] I \,. \tag{3.22}$$

If the beat period is less than the Ψ temporal resolution window, the beats are *filtered* out by the auditory brain (the effective ρ_{mn} is small) and we do not hear the beats. In this case $\Delta I(I) = \alpha_*^2 I$. This model needs to be tested [3.139].

Internal Noise. It is widely accepted that the pure-tone intensity JND is determined by the *internal noise* of the auditory system [3.140, 141], and that ΔI is proportional to the standard deviation of the Ψ-domain decision variable that is being discriminated in the intensity detection task, reflected back into the Φ domain. The usual assumption, from signal detection theory, is that $\Delta I = d'\sigma_I$, where d' is defined as the proportionality between the change in intensity and the variance $d' \equiv \Delta I/\sigma_I$. Threshold is typically when $d' = 1$ but can

depend on the the experimental design; σ_I is the intensity standard deviation of the Φ-domain intensity due to Ψ-domain auditory noise [3.15, 17, 127].

Hearing Threshold. The *hearing threshold* (or unmasked threshold) *intensity* may be defined as the intensity corresponding to the first (lowest intensity) JND. The hearing threshold is represented as $I_p^*(0)$ to indicate the probe intensity when the masker intensity is small (i. e., $I \to 0$). It is believed that internal noise is responsible for the hearing threshold.

Loudness L. The *loudness* L of a sound is the Ψ intensity. The *loudness growth function* $L(I)$ depends on the stimulus conditions. For example $L(I)$ for a tone and for wide-band noise are not the same functions. Likewise the loudness growth function for a 100 ms tone and a 1 s tone differ. When defining a *loudness scale* it is traditional to specify the intensity, frequency, and duration of a tone such that the loudness growth function is one [$L(I_{ref}, f_{ref}, T_{ref}) = 1$ defines a loudness scale]. For the sone scale, the reference signal is a $I_{ref} = 40\,\text{dB} - \text{SPL}$ tone at $f_{ref} = 1\,\text{kHz}$ with duration $T_{ref} = 1\,\text{s}$. For Fletcher's LU scale the reference intensity is the hearing threshold, which means that 1 sone = 975 LU [3.42] for a normal hearing person. Fletcher's LU loudness scale seems a more-natural scale than the sone scale used in the American National Standards Institute (ANSI) and International Organization for Standardization (ISO) standards.

The Single-Trial Loudness. A fundamental postulate of psychophysics is that all decision variables (i. e., Ψ variables) are random variables, drawn from some probability space [3.132, Chap. 5]. For early discussions of this point see *Montgomery* [3.142] and p. 144 of *Stevens* and *Davis* [3.137]. To clearly indicate the distinction between random and nonrandom variables, a tilde ($\sim$) is used to indicate a random variable. As a mnemonic, we can think of the $\sim$ as a *wiggle* associated with randomness.

We define the loudness decision variable as the *single-trial loudness* $\widetilde{L}$, which is the sample loudness heard on each stimulus presentation. The loudness L is then the expected value of the single-trial loudness $\widetilde{L}$

$$L(I) \equiv \mathcal{E}\widetilde{L}(I)\,. \tag{3.23}$$

The second moment of the single-trial loudness

$$\sigma_L^2 \equiv \mathcal{E}(\widetilde{L} - L)^2 \tag{3.24}$$

defines the loudness *variance* σ_L^2 and *standard deviation* σ_L.

Derived Definitions

The definitions given above cover the basic variables. However many alternative forms (various normalizations) of these variables are used in the literature. These derived variables were frequently formed with the hope of finding an invariance in the data. This could be viewed as a form of modeling exercise that has largely failed (e.g., the near miss to Weber's law), and the shear number of combinations has led to serious confusions [3.138, p. 152]. Each normalized variable is usually expressed in dB, adding an additional unnecessary layer of confusion to the picture. For example, *masking* is defined as the masked threshold normalized by the unmasked (quiet) threshold, namely

$$M \equiv \frac{I_p^*(I_m)}{I_p^*(0)}\,.$$

It is typically quoted in dB re sensation level (dB − SL). The intensity JND is frequently expressed as a *relative JND* called the *Weber fraction* defined by

$$J(I) \equiv \frac{\Delta I(I)}{I}\,. \tag{3.25}$$

From the signal detection theory premise that $\Delta I = d'\sigma_I$ [3.17], J is just the reciprocal of an effective signal-to-noise ratio defined as

$$\text{SNR}_I(I) \equiv \frac{I}{\sigma_I(I)} \tag{3.26}$$

since

$$J = d'\frac{\sigma_I}{I} = \frac{d'}{\text{SNR}_I}\,. \tag{3.27}$$

One conceptual difficulty with the Weber fraction J is that it is an *effective* signal-to-noise ratio, expressed in the Φ (physical) domain, but determined by a Ψ (psychophysical) domain mechanism (internal noise), as may be seen from Fig. 3.11.

Loudness JND ΔL. Any suprathreshold Ψ-domain increments may be quantified by corresponding Φ domain increments. The *loudness* JND $\Delta L(I)$ is defined as the change in loudness $L(I)$ corresponding to the intensity JND $\Delta I(I)$. While it is not possible to measure ΔL directly, we assume that we may expand the loudness function in a Taylor series (Fig. 3.11), giving

$$L(I + \Delta I) = L(I) + \Delta I \left.\frac{dL}{dI}\right|_I + \text{HOT}\,,$$

where HOT represents *higher-order terms*, which we shall ignore. If we solve for

$$\Delta L \equiv L(I + \Delta I) - L(I) \tag{3.28}$$

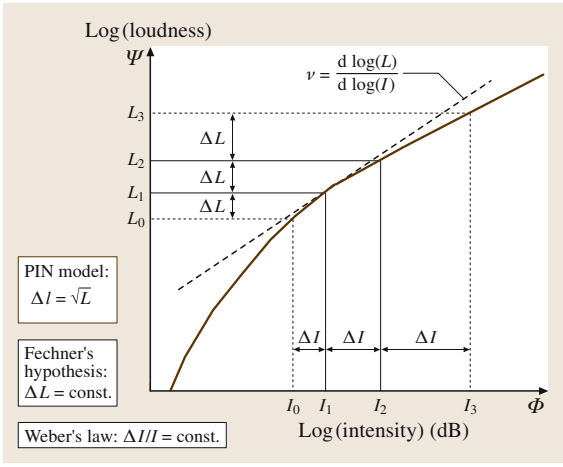


Fig. 3.11 Summary of all historical ideas about psychophysics and the relations between the Φ and Ψ variables. Along the abscissa we have the physical variable, intensity, and along the ordinate, the psychological variable loudness. The *curve* represents the loudness, on a log-intensity log-loudness set of scales. A JND in loudness is shown as ΔL and it depends on loudness, as described by the *Poisson internal noise* (PIN) model shown in the box on the left. Fechner assumed that ΔL was constant, which we now know to be incorrect. The loudness JND is reflected back into the physical domain as an intensity JND ΔI, which also depends on level. Weber's law, is therefore not true in general (but is approximately true for wide-band noise). Our analysis shows that the loudness SNR and the intensity SNR must be related by the slope of the loudness growth function, as given by (3.32). These relations are verified in Fig. 3.12, as discussed in detail in *Allen* and *Neely* [3.127]

we find

$$\Delta L = \Delta I \left.\frac{\mathrm{d}L}{\mathrm{d}I}\right|_{\mathrm{I}} . \tag{3.29}$$

We call this expression the *small*-JND approximation. The above shows that the loudness JND $\Delta L(I)$ is related to the intensity JND $\Delta I(I)$ by the slope of the loudness function, evaluated at intensity I. According to the signal detection model, the standard deviation of the single-trial loudness is proportional to the loudness JND, namely

$$\Delta L = d' \sigma_{\mathrm{L}} . \tag{3.30}$$

A more explicit way of expressing this assumption is

$$\frac{\Delta L}{\Delta I} = \frac{\sigma_{\mathrm{L}}}{\sigma_{\mathrm{I}}} , \tag{3.31}$$

where d' in both the Φ and Ψ domains is the same and thus cancels.

Loudness SNR. In a manner analogous to the Φ-domain $\mathrm{SNR}_{\mathrm{I}}$, we define the Ψ-domain loudness SNR as $\mathrm{SNR}_{\mathrm{L}}(L) \equiv L/\sigma_{\mathrm{L}}(L)$. Given (3.30), it follows that

$$\mathrm{SNR}_{\mathrm{I}} = \nu \mathrm{SNR}_{\mathrm{L}} , \tag{3.32}$$

where ν is the slope of the log-loudness function with respect to log-intensity. If we express the loudness as a power law

$$L(I) = I^{\nu}$$

and let $x = \log(I)$ and $y = \log(L)$, then $y = \nu x$. If the change of ν with respect to dB – SPL is small, then $\mathrm{d}y/\mathrm{d}x \approx \Delta y/\Delta x \approx \nu$. Since $\mathrm{d}\log(y) = \mathrm{d}y/y$ we get

$$\frac{\Delta L}{L} = \nu \frac{\Delta I}{I} . \tag{3.33}$$

Equation (3.32) is important because

1. it tells us how to relate the SNRs between the Φ and Ψ domains,
2. every term is dimensionless,
3. the equation is simple, since $\nu \approx 1/3$ is approximately constant above 40 dB – SL (i. e., Stevens' law), and because
4. we are used to seeing and thinking of loudness, intensity, and the SNR, on log scales, and ν as the slope on log–log scales.

Counting JNDs. While the concept of counting JNDs has been frequently discussed in the literature, starting with Fechner, unfortunately the actual counting formula (i. e., the equation) is rarely provided. As a result of a literature search, we found the formula in *Nutting* [3.143], *Fletcher* [3.21], *Wegel* and *Lane* [3.20], *Riesz* [3.75], *Fletcher* [3.144], and *Miller* [3.133].

To derive the JND counting formula, (3.29) is rewritten as

$$\frac{\mathrm{d}I}{\Delta I} = \frac{\mathrm{d}L}{\Delta L} . \tag{3.34}$$

Integrating over an interval gives the total number of intensity JNDs

$$N_{12} \equiv \int_{I_1}^{I_2} \frac{\mathrm{d}I}{\Delta I} = \int_{L_1}^{L_2} \frac{\mathrm{d}L}{\Delta L} , \tag{3.35}$$

where $L_1 = L(I_1)$ and $L_2 = L(I_2)$. Each integral counts the total number of JNDs in a different way between I_1 and I_2 [3.75,144]. The number of JNDs must be the same regardless of the domain (i. e., the abscissa variable), Φ or Ψ.

3.3.2 Empirical Models

This section reviews some earlier empirical models of the JND and its relation to loudness relevant to our development.

Weber's Law

In 1846 it was suggested by Weber that $J(I)$ is independent of I. According to (3.21) and (3.25)

$$J(I) = 2\alpha_* + \alpha_*^2 \,.$$

If J is constant, then α_* must be constant, which we denote by $\alpha_*(\not{I})$ (we strike out I to indicate that α_* is not a function of intensity). This expectation, which is called Weber's law [3.145], has been successfully applied to many human perceptions. We refer the reader to the helpful and detailed review of these questions by *Viemeister* [3.129], *Johnson* et al. [3.146], and *Moore* [3.147].

Somewhat frustrating is the empirical observation that $J(I)$ is not constant for the most elementary case of a pure tone [3.75, 136]. This observation is referred to as *the near miss to Weber's law* [3.148].

Weber's law does make one simple prediction that is potentially important. From (3.35) along with Weber's law $J_0 \equiv J(\not{I})$ we see that the formula for the number of JNDs is

$$N_{12} = \int_{I_1}^{I_2} \frac{\mathrm{d}I}{J_0 I} = \frac{1}{J_0} \ln\left(\frac{I_2}{I_1}\right) . \tag{3.36}$$

It remains unexplained why Weber's law holds as well as it does [3.149, 150, p. 721] (it holds approximately for the case of wide band noise), or even why it holds at all. Given the complex and NL nature of the transformation between the Φ and Ψ domains, coupled with the belief that the noise source is in the Ψ domain, it seems unreasonable that a law as simple as Weber's law could hold in any general way. A transformation of the JND from the Φ domain to the Ψ domain greatly clarifies the situation.

Fechner's Postulate

In 1860 Fechner postulated that the loudness JND $\Delta L(I)$ is a constant [3.125, 130, 151, 152]. We are only considering the auditory case of Fechner's more general theory. We shall indicate such a constancy with respect to I as $\Delta L(\not{I})$ (as before, we strike out the I to indicate that ΔL is *not* a function of intensity). As first reported by *Stevens* [3.153], we shall show that Fechner's postulate is not generally true.

The Weber–Fechner Law

It is frequently stated [3.152] that Fechner's postulate ($\Delta L(\not{I})$) and Weber's law ($J_0 \equiv J(\not{I})$) lead to the conclusion that the difference in loudness between any two intensities I_1 and I_2 is proportional to the logarithm of the ratio of the two intensities, namely

$$\frac{L(I_2) - L(I_1)}{\Delta L} = \frac{1}{J_0} \log\left(\frac{I_2}{I_1}\right) . \tag{3.37}$$

This is easily seen by eliminating N_{12} from (3.36) and by assuming Weber's law and Fechner's hypothesis. This result is called *Fechner's law* (also called the *Weber–Fechner law*). It is not true because of the faulty assumptions, Weber's law and Fechner's postulate.

3.3.3 Models of the JND

Starting in 1923, *Fletcher* and *Steinberg* studied loudness coding of pure tones, noise, and speech [3.21, 154–156], and proposed that loudness was related to neural spike count [3.41], and even provided detailed estimates of the relation between the number of spikes and the loudness in sones [3.42, p. 271]. In 1943 *De Vries* first introduced a photon-counting Poisson process model as a theoretical basis for the threshold of vision [3.157]. *Siebert* [3.140] proposed that Poisson point-process noise, resulting from the neural rate code, acts as the internal noise that limits the frequency JND [3.136, 150]. A few years later [3.158], and independently [3.159] *McGill* and *Goldberg* [3.160] proposed that the Poisson internal noise (PIN) model might account for the intensity JND, but they did not find this to produce a reasonable loudness growth function. *Hellman* and *Hellman* [3.134] further refined the argument that Poisson noise may be used to relate the loudness growth to the intensity JND, and they found good agreement between the JND and realistic loudness functions.

Given Poisson noise, the variance is equal to the mean, thus

$$\Delta L(L) \propto \sqrt{L} \,. \tag{3.38}$$

This may also be rewritten as $\sigma_L^2 \propto L$. We would expect this to hold if the assumptions of *McGill* [3.148] (i. e., the PIN model) are valid.

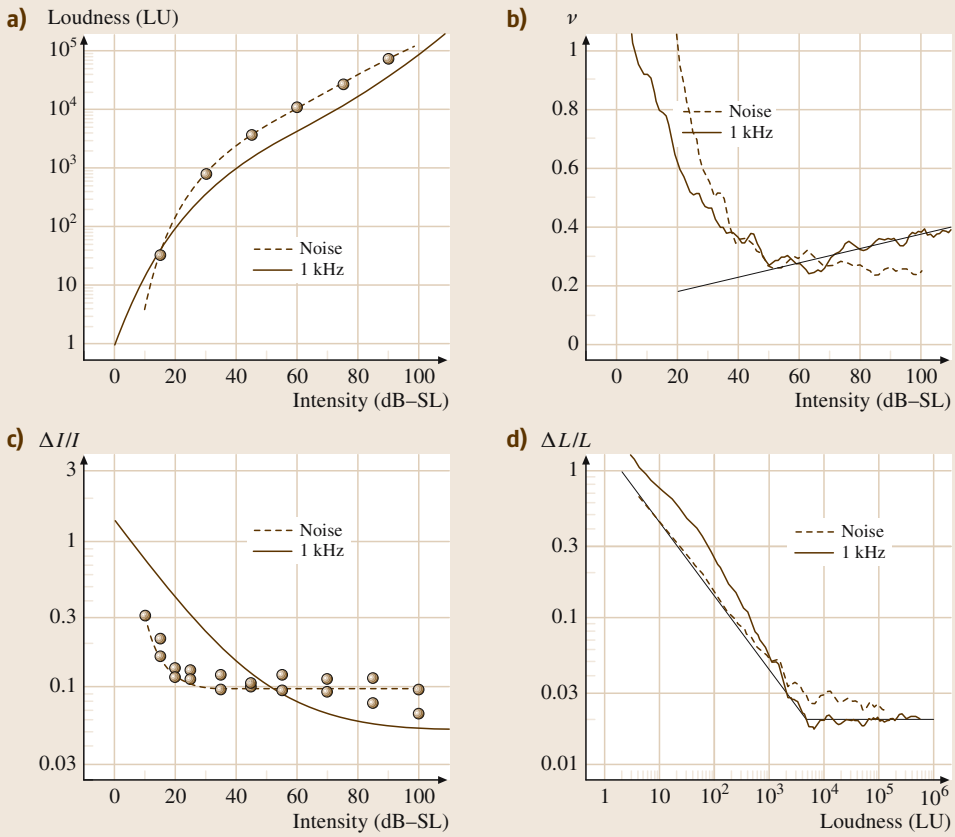


Fig. 3.12a–d In 1947 Miller measured the JND_I and the loudness level for two subjects using wide-band modulated noise (0.15–7 kHz) for levels between 3 and 100 dB − SL. The noise (*dashed line*) and pure tone (*solid line*) loudness are shown in **(a)**. The similarity between $\Delta L/L$ derived from the loudness curves for pure tones and for noise provide an almost perfect fit to the SPIN model which results from assuming the noise is neural point-process noise. See the text for a summary of these results. The direct derivation of ΔL based on pure tone JND and loudness data from *Miller* [3.133], *Riesz* [3.75], *Fletcher* and *Munson* [3.41].

In the following we directly compare the loudness–growth function of Fletcher and Munson to the number of JNDs N_{12} from Riesz [3.75, 127] to estimate $\Delta L/L$.

3.3.4 A Direct Estimate of the Loudness JND

Given its importance, it is important to estimate ΔL directly from its definition (3.28), using Riesz's $\Delta I(I)$ and Fletcher and Munson's 1933 estimate of $L(I)$.

Miller's 1947 famous JND paper includes wide-band-noise loudness-level results. We transformed these JND data to loudness using *Fletcher* and *Munson* [3.41] reference curve (i. e., Fig. 3.12a).

Loudness Growth, Recruitment, and the OHC

In 1924 *Fletcher* and *Steinberg* published an important paper on the measurement of the loudness of speech signals [3.155]. In this paper, when describing the growth of loudness, the authors state

> *the use of the above formula involved a summation of the cube root of the energy rather than the energy.*

This cube–root dependence had first been described by *Fletcher* the year before [3.21].

In 1930 *Fletcher* [3.27] postulated that there was a monotonic relationship between central nerve firings rates and loudness. Given a tonal stimulus at the ear

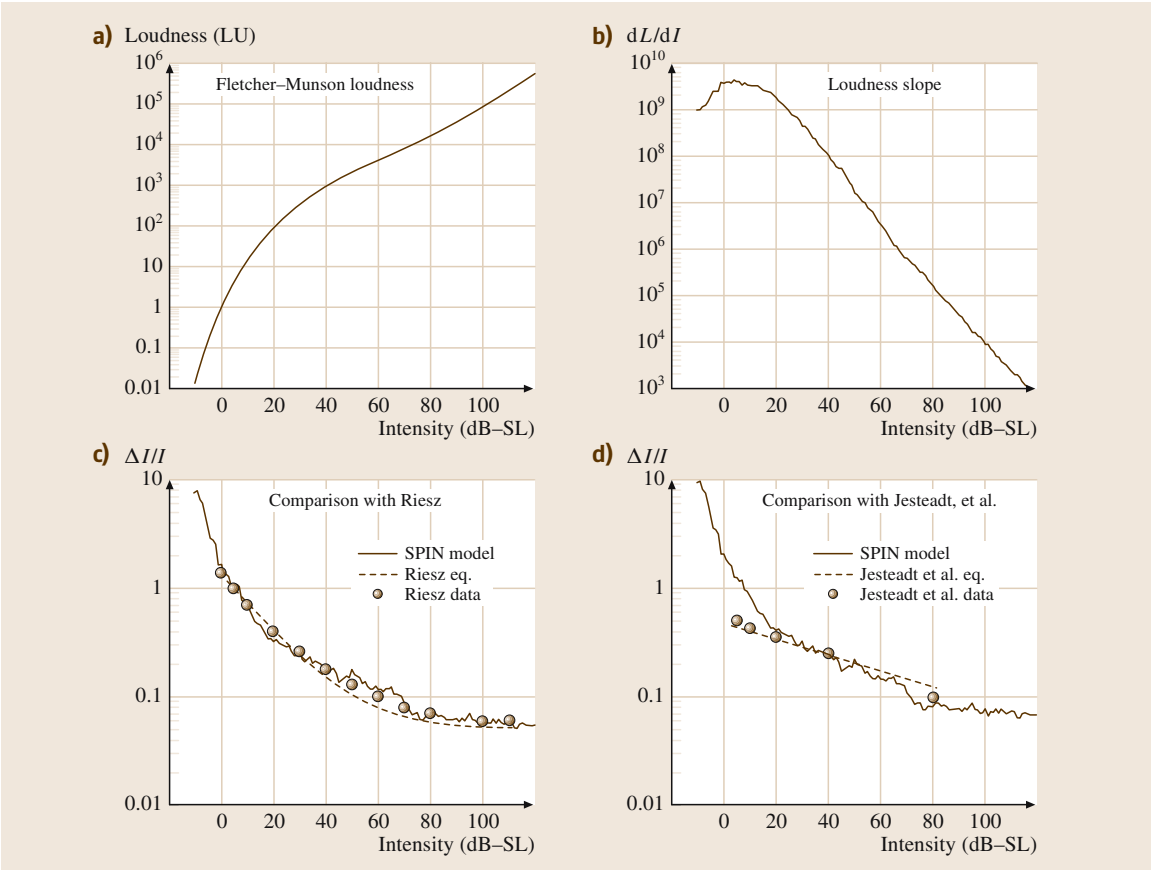


Fig. 3.13a–d Test of the SPIN model against the classic results of *Riesz* [3.75], *Jesteadt* et al. [3.136]. Test of the model derived on the *left* based on a comparison between loudness data and intensity JND data at 1 kHz, using the SPIN model

drum, Stevens' law says that the loudness is given by

$$L \equiv L(f, x, I) \propto I^{\nu} , \qquad (3.39)$$

where (f, x, I) are the frequency, place, and intensity of the tone, respectively. The exponent ν has been experimentally established to be in the range between 1/4 and 1/3 for long duration pure tones at 1 kHz. *Fletcher* and *Munson* [3.41] found $\nu \approx 1/4$ at high intensities and approximately 1 near threshold. Although apparently it has not been adequately documented, ν seems to be close to 1 for the *recruiting ear* [3.15].

Recruitment. What is the source of Fletcher's cube-root loudness growth (i. e., Stevens' law)? Today we know that cochlear outer hair cells are the source of the cube-root loudness growth observed by Fletcher.

From noise trauma experiments on animals and humans, we may conclude that recruitment (abnormal loudness growth) occurs in the cochlea [3.3, 96]. *Steinberg* and *Gardner* described such a loss as a *variable loss* (i. e., sensory neural loss) and partial recruitment as a mixed loss (i. e., having a conductive component) [3.3, 161]. They and Fowler verified the conductive component by estimating the air–bone gap. In a comment to Fowler's original presentation on loudness recruitment in 1937, the famous anatomist *Lorente de Nó* theorized that recruitment is due to hair cell damage [3.14]. *Steinberg* and *Gardner* clearly understood recruitment, as is indicated in the following quote [3.3, p. 20]

> *Owing to the expanding action of this type of loss it would be necessary to introduce a corresponding compression in the amplifier in order to produce the same amplification at all levels.*

This compression/loss model of hearing and hearing loss, along with the loudness models of *Fletcher* and *Munson* [3.41], are basic to an eventual quantitative understanding of NL cochlear signal processing and the cochlea's role in detection, masking and loudness in normal and impaired ears. The work by *Fletcher* [3.162] and *Steinberg* and *Gardner* [3.3], and work on modeling hearing loss and recruitment [3.122] support this view.

In summary, many studies conclude that the cube-root loudness growth starts with the NL compression of basilar membrane motion due to stimulus-dependent voltage changes within the OHC.

3.3.5 Determination of the Loudness SNR

In Fig. 3.12 we show a summary of $L(I)$, $\nu(I)$, $J(I)$ and $\Delta L/L = d'/\mathrm{SNR_L}$ for the tone and noise data.

The pure-tone and wide-band noise JND results may be summarized in terms of the loudness $\mathrm{SNR_L}(L)$ data shown in Fig. 3.12d where we show $\Delta L/L = d'/\mathrm{SNR_L}$, as a function of loudness.

For noise below $55\,\mathrm{dB} - \mathrm{SL}$ ($L < 5000\,\mathrm{LU}$) the loudness signal-to-noise ratio $\mathrm{SNR_L} \equiv L/\sigma_\mathrm{L}$ decreases as the square root of the loudness. For a loudness greater than $5000\,\mathrm{LU}$ ($N \approx 5$ sones), $\Delta L/L \approx 0.025$ fn both tones and noise (Fig. 3.12d)

In the lower-right panel (Fig. 3.12d) we provide a functional summary of $\Delta L/L$ for both tones and noise with the light solid line described by

$$\frac{\Delta L(L)}{L} = h\,[\min(L, L_0)]^{-1/2}\ , \tag{3.40}$$

where $h = \sqrt{2}$ and $L_0 = 5000\,\mathrm{LU}$ (≈ 5 sone). We call this relation the *saturated Poisson internal noise* (SPIN) model. With these parameter values, (3.40) appears to be a lower bound on the relative loudness $\mathrm{JND_L}$ for both tones and noise. From (3.33) $\Delta L/L = \nu(I)J(I)$. Note how the product of $\nu(I)$ and $J(I)$ is close to a constant for tones above 5000 LU.

In Fig. 3.12b the second top panel shows the exponent $\nu(I)$ for both Fletcher and Munson's and Miller's loudness growth function. In the lower-left panel (Fig. 3.12c) we see $\Delta I/I$ versus I for Miller's subjects, Miller's equation, and Riesz's JND equation.

Near miss to Stevens' Law

For tones the intensity exponent $\nu(I)$ varies systematically between 0.3 and 0.4 above $50\,\mathrm{dB} - \mathrm{SL}$, as shown by the solid line in the upper-right panel of Fig. 3.12b. We have highlighted this change in the power law with intensity for a 1 kHz tone in the upper-right panel with a light solid straight line. It is logical to call this effect the *near miss to Stevens' law,* since it cancels the near miss to Weber's law, giving a constant relative loudness JND $\Delta L/L$ for tones.

Figure 3.13a shows the Fletcher–Munson loudness data from Table III in [3.41]. The upper-right panel (Fig. 3.13b) is the slope of the loudness with respect to intensity ($\mathrm{LU\,cm^2/W}$).In the lower–right (Fig. 3.13d) we compare the SPIN model relative JND (3.43) (with $h = 3.0$), and the relative JND computed from the *Jesteadt* et al. [3.136] formula (dashed line) and data from their Table B-I (circles). They measured the JND using pulsed tones for levels between 5 and 80 dB. The Jesteadt et al. data were taken with gated stimuli (100% modulation) and 2AFC methods. It is expected that the experimental method would lead to a different value of h than the valued required for Riesz's data set. The discrepancy between 0 and 20 dB may be due to the 100% modulation for these stimuli. The fit from 20 to $80\,\mathrm{dB} - \mathrm{SL}$ is less than a 5% maximum error, and much less in terms of RMS error. Note the similarity in slope between the model and the data.

3.3.6 Weber–Fraction Formula

In this section we derive the relation between the Weber fraction $J(I)$ given the loudness $L(I)$ starting from the *small-JND approximation*

$$\Delta L = \Delta I L'(I)\,, \tag{3.41}$$

where $L'(I) \equiv \mathrm{d}L/\mathrm{d}I$. If we solve this equation for ΔI and divide by I we find

$$J(I) \equiv \frac{\Delta I}{I} = \frac{\Delta L}{IL'(I)}\ . \tag{3.42}$$

Finally we substitute the SPIN model (3.40)

$$J(I) = \frac{hL(I)}{IL'(I)}\,[\min(L(I), L_0)]^{-1/2}\ . \tag{3.43}$$

This formula is the same as that derived by *Hellman* and *Hellman* [3.134], when $L \le L_0$. In Fig. 3.13c we plot (3.43) labeled *SPIN-model* with $h = 2.4$ and $L_0 = 10\,000\,\mathrm{LU}$. For levels between 0 and $100\,\mathrm{dB} - \mathrm{SL}$, the SPIN model (solid curve) fit to Riesz's data and Riesz's formula is excellent. Over this 100 dB range the curve defined by the loudness function fits as well as the curve defined by Riesz's formula [3.127]. The excellent fit gives us further confidence in the basic assumptions of the model.

3.4 Discussion and Summary

Inspired by the Poisson internal noise (PIN)-based theory of *Hellman* and *Hellman* [3.134], we have developed a theoretical framework that can be used to explore the relationship between the pure-tone loudness and the intensity JND. The basic idea is to combine Fletcher's neural excitation response pattern model of loudness with signal detection theory. We defined a random decision variable called the single-trial loudness. The *mean* of this random variable is the loudness, while its *standard deviation* is proportional to the loudness JND. We define the loudness signal-to-noise ratio $\mathrm{SNR_L}$ as the ratio of loudness (the signal) to standard deviation (a measure of the noise).

3.4.1 Model Validation

To evaluate the model we have compared the loudness data of *Fletcher* and *Munson* [3.41] with the intensity JND data of *Riesz* [3.75], for tones. A similar comparison was made for noise using loudness and intensity JND data from *Miller* [3.133]. We were able to unify the tone and noise data by two equivalent methods in Fig. 3.12d. Since the loudness SNR is proportional to the ratio of the loudness to the JND $L/\Delta L$, the SNR is also a piecewise power-law function which we call the SPIN model. All the data are in excellent agreement with the SPIN model, providing support for the validity of this theory.

The above discussion has

- drawn out the fundamental nature of the JND,
- shown that the PIN loudness model holds below 5 sone (5000 LU) (the solid line in the lower right panel of Fig. 3.11 below 5000 LU obeys the PIN model, and the data for both tones and wide band noise fall close to this line below 5000 LU) (one sone is 975 LU [3.127, p. 3631], thus 5000 LU = 5.13 LU. From the loudness scale this corresponds to a 1 kHz pure tone at 60 dB − SL),
- shown that above 5 sone the PIN model fails and the loudness SNR remains constant.

3.4.2 The Noise Model

The SPIN Model

Equation (3.40) summarizes our results on the relative loudness JND for both tones and noise. Using this formula along with (3.32), the JND may be estimated for tones and noise once the loudness has been determined, by measurement, or by model. Fechner's postulate, that the loudness JND is constant, is not supported by our analysis, in agreement with *Stevens* [3.153].

The PIN Model

The success of the PIN model is consistent with the idea that the pure-tone loudness code is based on neural discharge rate. This theory should apply between threshold and moderate intensities (e.g., < 60 dB) for *frozen stimuli* where the JND is limited by internal noise.

CNS Noise

Above 60 dB − SL we find that the loudness signal-to-noise ratio saturated (Fig. 3.12d) with a constant loudness SNR between 30 and 50 for both the tone and noise conditions, as summarized by Ekman's law [3.163]. We conclude that the Hellman and Hellman theory must be modified to work above 5 sones.

Weber's Law

It is significant that, while both $J(I)$ and $\nu(I)$ vary with intensity, the product is constant above 60 dB − SL. Given that $J = d'/\nu\mathrm{SNR_L}$, the saturation in $\mathrm{SNR_L}$ explains Weber's law for wideband signals (since ν and $\mathrm{SNR_L}$ for that case are constant) as well as the near miss to Weber's law for tones, where ν is not constant (the near miss to Stevens' law, Fig. 3.12a).

Generalization to Other Data

If $\sigma_L(L, I)$ depends on L, and is independent of I, then the $\mathrm{SNR_L}(L)$ should not depend on the nature of the function $L(I)$ (i. e., it should be true for any $L(I)$). This prediction is supported by our analysis summarized by (3.40). It will be interesting to see how $\mathrm{SNR_L}$ depends on L and I for subjects having a hearing-loss-induced recruitment, and how well this theory explains other data in the literature, such as loudness and JNDs with masking-induced recruitment [3.126].

Conditions for Model Validity

To further test the SPIN model, several conditions must be met. First the loudness and the JND must have been measured under the same stimulus conditions. Second, the internal noise must be the dominant factor in determining the JND. This means that the stimuli must be frozen (or have significant duration and bandwidth), and the subjects well trained in the task. As the signal uncertainty begins to dominate the internal noise, as it does in

the cases of roving the stimulus, the intensity JND will become independent of the loudness.

As discussed by *Stevens* and *Davis* [3.164, pp. 141-143], JND data are quite sensitive to the modulation conditions. The *Riesz* [3.75] and *Munson* [3.165] data make an interesting comparison because they are taken under steady–state conditions and are long duration tonal signals. Both sets of experimental data (i. e., Riesz and Munson) were taken in the same laboratory within a few years of each other. In 1928 Wegel, Riesz, and Munson were all members of Fletcher's department. *Riesz* [3.75] states that he used the same methods as *Wegel* and *Lane* [3.20], and it is likely that *Munson* [3.165] did as well.

Differences in the signal conditions are the most likely explanation for the differences observed in the intensity JND measurements of Riesz and Jesteadt shown in Fig. 3.13d. One difference between the data of *Riesz* [3.75] and *Jesteadt* et al. [3.136] is that Riesz varied the amplitude of the tones in a sinusoidal manner with a small (i. e., just detectable) modulation index, while Jesteadt et al. alternated between two intervals of different amplitude, requiring that the tones be gated on and off (i. e., a 100% modulation index).

The neural response to transient portions of a stimulus is typically larger than the steady-state response (e.g., neural overshoot) and, therefore, may dominate the perception of stimuli with large, abrupt changes in amplitude. The fact that the intensity JND is sensitive to the time interval between two tones of different amplitude [3.164] is another indication that neural overshoot may play a role.

It would be interesting to check the SPIN model on loudness and JND data taken using gated signals, given the observed sensitivity to the modulation. While these JND data are available [3.136], one would need loudness data taken with identical (or at least similar) modulations. We are not aware of such data.

References

3.1 E. Relkin, C. Turner: A reexamination of forward masking in the auditory nerve, J. Acoust. Soc. Am. **84**(2), 584–591 (1988)

3.2 M. Hewitt, R. Meddis: An evaluation of eight computer models of mammalian inner hair-cell function, J. Acoust. Soc. Am. **90**(2), 904–917 (1991)

3.3 J. Steinberg, M. Gardner: Dependence of hearing impairment on sound intensity, J. Acoust. Soc. Am. **9**, 11–23 (1937)

3.4 J.O. Pickles: *An Introduction to the Physiology of Hearing* (Academic, London 1982)

3.5 P. Dallos: Cochlear neurobiology. In: *The Cochlea*, ed. by P. Dallos, A. Popper, R. Fay (Springer, New York 1996) pp. 186–257

3.6 W.A. Yost: *Fundamentals of Hearing, An Introduction* (Academic, San Diego, London 2006)

3.7 S. Hecht: Vision II, The nature of the photoreceptor process. In: *Handbook of General Experimental Psychology*, ed. by C. Murchison (Clark Univ. Press, Worcester 1934)

3.8 G. Gescheider: *Psychophysics: The Fundamentals*, 3rd edn. (Lawrence Erlbaum, Mahwah 1997)

3.9 J.B. Allen, S. Neely: Micromechanical models of the cochlea, Phys. Today **45**(7), 40–47 (1992)

3.10 J.B. Allen: Harvey Fletcher's role in the creation of communication acoustics, J. Acoust. Soc. Am. **99**(4), 1825–1839 (1996)

3.11 A. Hudspeth, D. Corey: Sensitivity, polarity, and conductance change in the response of vertebrate hair cells to controlled mechanical stimuli, Proc. Nat. Acad. Sci. **74**(6), 2407–2411 (1977)

3.12 M. Liberman: Single-neuron labeling in the cat auditory nerve, Science **216**, 163–176 (1982)

3.13 J. Steinberg: Stereophonic sound-film system-pre- and post-equalization of compandor systems, J. Acoust. Soc. Am. **13**, 107–114 (1941), B-1327

3.14 R.L. de Nó: The diagnosis of diseases of the neural mechanism of hearing by the aid of sounds well above threshold, T. Am. Otol. Soc. **27**, 219–220 (1937looseness1)

3.15 S.T. Neely, J.B. Allen: Relation between the rate of growth of loudness and the intensity DL. In: *Modeling Sensorineural Hearing Loss*, ed. by W. Jesteadt (Lawrence Erlbaum, Mahwah 1997) pp. 213–222

3.16 T.S. Littler: *The Physics of the Ear* (Pergamon, Oxford 1965)

3.17 W.M. Hartmann: *Signals, Sound, and Sensation* (AIP Press, Woodbury 1997)

3.18 H.L.F. Helmholtz: *On the Sensations of Tone* (Dover, New York 1954), 1863

3.19 H.L.F. Helmholtz: *Helmholtz's popular scientific lectures* (Dover, New York 1962), 1857

3.20 R.L. Wegel, C. Lane: The auditory masking of one pure tone by another and its probable relation to the dynamics of the inner ear, Phys. Rev. **23**, 266–285 (1924)

3.21 H. Fletcher: Physical measurements of audition and their bearing on the theory of hearing, J. Franklin Inst. **196**(3), 289–326 (1923)

3.22 G. Campbell: On loaded lines in telephonic transmission, Phil. Mag. **5**, 313–331 (1903)

3.23 G. Campbell: Physical theory of the electric wave filter, Bell System Tech. J. **1**(1), 1–32 (1922)
3.24 G.A. Campbell: Telephonic intelligibility, Phil. Mag. **19**(6), 152–159 (1910)
3.25 H. Fletcher: The nature of speech and its interpretation, J. Franklin Inst. **193**(6), 729–747 (1922)
3.26 H. Fletcher, J. Steinberg: Articulation testing methods, J. Acoust. Soc. Am. **1**(2.2), 17–113 (1930), Intelligibility Lists pp. 65–113)
3.27 H. Fletcher: A space-time pattern theory of hearing, J. Acoust. Soc. Am. **1**(1), 311–343 (1930)
3.28 J. Zwislocki: Theorie der Schneckenmechanik, Acta Otolaryngol. **72**, 1–112 (1948)
3.29 J. Zwislocki: Theory of the acoustical action of the cochlea, J. Acoust. Soc. Am. **22**, 779–784 (1950)
3.30 O. Ranke: Theory operation of the cochlea: A contribution to the hydrodynamics of the cochlea, J. Acoust. Soc. Am. **22**, 772–777 (1950)
3.31 L.C. Peterson, B.P. Bogert: A dynamical theory of the cochlea, J. Acoust. Soc. Am. **22**, 369–381 (1950)
3.32 H. Fletcher: On the dynamics of the cochlea, J. Acoust. Soc. Am. **23**, 637–645 (1951)
3.33 H. Fletcher: Acoustics, Phys. Today **4**, 12–18 (1951)
3.34 S. Puria, J.B. Allen: Measurements and model of the cat middle ear: Evidence for tympanic membrane acoustic delay, J. Acoust. Soc. Am. **104**(6), 3463–3481 (1998)
3.35 J.B. Allen, P.S. Jeng, H. Levitt: Evaluating human middle ear function via an acoustic power assessment, J. Rehabil. Res. Dev. **42**(4), 63–78 (2005)
3.36 H. Fletcher, W. Munson: Relation between loudness and masking, J. Acoust. Soc. Am. **9**, 1–10 (1937)
3.37 H. Fletcher: Loudness, masking and their relation to the hearing process and the problem of noise measurement, J. Acoust. Soc. Am. **9**, 275–293 (1938)
3.38 H. Fletcher: Auditory patterns, Rev. Mod. Phys. **12**(1), 47–65 (1940)
3.39 J. Steinberg: Positions of stimulation in the cochlea by pure tones, J. Acoust. Soc. Am. **8**, 176–180 (1937), cochlear map estimate, Monograph B-973
3.40 D.D. Greenwood: Auditory masking and the critical band, J. Acoust. Soc. Am. **33**(4), 484–502 (1961)
3.41 H. Fletcher, W. Munson: Loudness, its definition, measurement, and calculation, J. Acoust. Soc. Am. **5**, 82–108 (1933)
3.42 H. Fletcher: *Speech and Hearing in Communication* (Krieger, Huntington 1953)
3.43 D.D. Greenwood: Critical bandwidth and the frequency coordinates of the basilar membrane, J. Acoust. Soc. Am. **33**, 1344–1356 (1961)
3.44 M. Liberman: The cochlear frequency map for the cat: Labeling auditory-nerve fibers of known characteristic frequency, J. Acoust. Soc. Am. **72**(5), 1441–1449 (1982)
3.45 M. Liberman, L. Dodds: Single neuron labeling and chronic cochlear pathology III: Stereocilia damage and alterations of threshold tuning curves, Hearing Res. **16**, 55–74 (1984)
3.46 W. Rhode: Some observations on cochlear mechanics, J. Acoust. Soc. Am. **64**, 158–176 (1978)
3.47 W. Rhode: Observations of the vibration of the basilar membrane in squirrel monkeys using the Mössbauer technique, J. Acoust. Soc. Am. **49**, 1218–1231 (1971)
3.48 P. Sellick, I. Russell: Intracellular studies of cochlear hair cells: Filling the gap between basilar membrane mechanics and neural excitation. In: *Evoked Electrical Activity in the Auditory Nervous System*, ed. by F. Naunton, C. Fernandez (Academic, New York 1978) pp. 113–140
3.49 D. Kemp: Stimulated acoustic emissions from within the human auditory system, J. Acoust. Soc. Am. **64**, 1386–1391 (1978)
3.50 W. Brownell, C. Bader, D. Bertran, Y. de Rabaupierre: Evoked mechanical responses of isolated cochlear outer hair cells, Science **227**, 194–196 (1985)
3.51 J.B. Allen: OHCs shift the excitation pattern via BM tension. In: *Diversity in Auditory Mechanics*, ed. by E. Lewis, G. Long, R. Lyon, P. Narins, C. Steele, E. Hecht-Poinar (World Scientific, Singapore 1997) pp. 167–175
3.52 J.L. Goldstein, N. Kiang: Neural correlates of the aural combination tone 2f1-f2, Proc. IEEE **56**, 981–992 (1968)
3.53 G. Smoorenburg: Combination tones and their origin, J. Acoust. Soc. Am. **52**(2), 615–632 (1972)
3.54 D. Kemp: Evidence of mechanical nonlinearity and frequency selective wave amplification in the cochlea, Arch. Oto-Rhino-Laryngol. **224**, 37–45 (1979)
3.55 D. Kim, J. Siegel, C. Molnar: Cochlear nonlinear phenomena in two-tone responses, Scand. Audiol. **9**, 63–82 (1979)
3.56 P.F. Fahey, J.B. Allen: Nonlinear phenomena as observed in the ear canal and at the auditory nerve, J. Acoust. Soc. Am. **77**(2), 599–612 (1985)
3.57 M.B. Sachs, Y.S. Kiang: Two-tone inhibition in auditory-nerve fibers, J. Acoust. Soc. Am. **43**, 1120–1128 (1968)
3.58 R. Arthur, R. Pfeiffer, N. Suga: Properties of "two-tone inhibition" in primary auditory neurons, J. Physiol. (London) **212**, 593–609 (1971)
3.59 N.-S. Kiang, E. Moxon: Tails of tuning curves of auditory-nerve fibers, J. Acoust. Soc. Am. **55**, 620–630 (1974)
3.60 P. Abbas, M. Sachs: Two-tone suppression in auditory–nerve fibers: Extension of a stimulus–response relationship, J. Acoust. Soc. Am. **59**(1), 112–122 (1976)
3.61 X. Pang, J. Guinan: Growth rate of simultaneous masking in cat auditory-nerve fibers: Relationship to the growth of basilar-membrane motion and the origin of two-tone suppression, J. Acoust. Soc. Am. **102**(6), 3564–3574 (1997), beautiful data showing the slope of suppression for low frequency suppressors

3.62 I. Russell, P. Sellick: Intracellular studies of hair cells in the mammalian cochlea, J. Physiol. **284**, 261–290 (1978)

3.63 H. Duifhuis: Consequences of peripheral frequency selectivity for nonsimultaneous masking, J. Acoust. Soc. Am. **54**(6), 1471–1488 (1973)

3.64 D. Kemp: The evoked cochlear mechanical response and the auditory microstructure – evidence for a new element in cochlear mechanics, Scand. Audiol. **9**, 35–47 (1979)

3.65 D. Kemp: Towards a model for the origin of cochlear echoes, Hearing Res. **2**, 533–548 (1980)

3.66 D. Kemp: Otoacoustic emissions, travelling waves and cochlear mechanisms, J. Acoust. Soc. Am. **22**, 95–104 (1986)

3.67 A. Recio, W. Rhode: Basilar membrane responses to broadband stimuli, J. Acoust. Soc. Am. **108**(5), 2281–2298 (2000)

3.68 E. Fowler: A method for the early detection of otosclerosis, Archiv. Otolaryngol. **24**(6), 731–741 (1936)

3.69 S. Neely, D.O. Kim: An active cochlear model showing sharp tuning and high sensitivity, Hearing Res. **9**, 123–130 (1983)

3.70 D. He, P. Dallos: Somatic stiffness of cochlear outer hair cells is voltage-dependent, Proc. Nat. Acad. Sci. **96**(14), 8223–8228 (1999)

3.71 D. He, P. Dallos: Properties of voltage-dependent somatic stiffness of cochlear outer hair cells, J. Assoc. Res. Otolaryngol. **1**(1), 64–81 (2000)

3.72 A.M. Mayer: Research in acoustics, Phil. Mag. **2**, 500–507 (1876), in Benchmark Papers in Acoustics, Vol. 13, edited by E. D. Schubert

3.73 E. Titchener: *Experimental Psychology, A Manual of Laboratory Practice*, Vol. II (Macmillan, London 1923)

3.74 J.B. Allen: A short history of telephone psychophysics, J. Audio Eng. Soc. Reprint **4636**, 1–37 (1997)

3.75 R.R. Riesz: Differential intensity sensitivity of the ear for pure tones, Phys. Rev. **31**(2), 867–875 (1928)

3.76 D. McFadden: The curious half–octave shift: Evidence of a basalward migration of the traveling–wave envelope with increasing intensity. In: *Applied and Basic Aspects of Noise-Induced Hearing Loss*, ed. by R. Salvi, D. Henderson, R. Hamernik, V. Coletti (Plenum, New York 1986) pp. 295–312

3.77 W.A. Munson, M.B. Gardner: Loudness patterns – a new approach, J. Acoust. Soc. Am. **22**(2), 177–190 (1950)

3.78 B. Strope, A. Alwan: A model of dynamic auditory perception and its application to robust word recognition, IEEE Trans. Acoust. Speech Signal Proc. **5**(5), 451–464 (1997)

3.79 J.B. Allen: Psychoacoustics. In: *Wiley Encyclopedia of Electrical and Electronics Engineering*, Vol. 17, ed. by J. Webster (Wiley, New York 1999) pp. 422–437

3.80 B. Delgutte: Two-tone suppression in auditory-nerve fibres: Dependence on suppressor frequency and level, Hearing Res. **49**, 225–246 (1990)

3.81 B. Delgutte: Physiological mechanisms of psychophysical masking: Observations from auditory-nerve fibers, J. Acoust. Soc. Am. **87**, 791–809 (1990)

3.82 B. Delgutte: Physiological models for basic auditory percepts. In: *Auditory Computation*, ed. by H. Hawkins, T. McMullen (Springer, New York 1995)

3.83 L. Kanis, E. de Boer: Two-tone suppression in a locally active nonlinear model of the cochlea, J. Acoust. Soc. Am. **96**(4), 2156–2165 (1994)

3.84 J. Hall: Two-tone distortion products in a nonlinear model of the basilar membrane, J. Acoust. Soc. Am. **56**, 1818–1828 (1974)

3.85 C.D. Geisler, A.L. Nuttall: Two-tone suppression of basilar membrane vibrations in the base of the guinea pig cochlea using "low-side" suppressors, J. Acoust. Soc. Am. **102**(1), 430–440 (1997)

3.86 M. Régnier, J.B. Allen: *The Importance of Across-Frequency Timing Coincidences in the Perception of Some English Consonants in Noise* (ARO, Denver 2007), Abstract

3.87 M. Régnier, J.B. Allen: *Perceptual Cues of Some CV Sounds Studied in Noise* (AAS, Scottsdale 2007), Abstract

3.88 S. Narayan, A. Temchin, A. Recio, M. Ruggero: Frequency tuning of basilar membrane and auditory nerve fibers in the same cochleae, Science **282**, 1882–1884 (1998), shows BM and neural differ by 3.8 dB/oct

3.89 E. deBoer: Mechanics of the cochlea: Modeling efforts. In: *The Cochlea*, ed. by P. Dallos, A. Popper, R. Fay (Springer, New York 1996) pp. 258–317

3.90 D.C. Geisler: *From Sound to Synapse: Physiology of the Mammalian Ear* (Oxford Univ. Press, Oxford 1998)

3.91 J.B. Allen, P.F. Fahey: Using acoustic distortion products to measure the cochlear amplifier gain on the basilar membrane, J. Acoust. Soc. Am. **92**(1), 178–188 (1992)

3.92 S. Neely, D.O. Kim: A model for active elements in cochlear biomechanics, J. Acoust. Soc. Am. **79**, 1472–1480 (1986)

3.93 E. deBoer, A. Nuttall: The mechanical waveform of the basilar membrane, III Intensity effects, J. Acoust. Soc. Am. **107**(3), 1496–1507 (2000)

3.94 D. Kim, S. Neely, C. Molnar, J. Matthews: An active cochlear model with negative damping in the cochlear partition: Comparison with Rhode's ante- and post-mortem results. In: *Psychological, Physiological and Behavioral Studies in Hearing*, ed. by G. van den Brink, F. Bilsen (Delft Univ. Press, Delft 1980) pp. 7–14

3.95 J.B. Allen: DeRecruitment by multiband compression in hearing aids. In: *Psychoacoustics, Speech, and Hearing aids*, ed. by B. Kollmeier (World Scientific, Singapore 1996) pp. 141–152

3.96 W.F. Carver: Loudness balance procedures. In: *Handbook of Clinical Audiology*, 2nd edn., ed. by J. Katz (Williams Wilkins, Baltimore 1978) pp. 164–178, Chap. 15

3.97 J.B. Allen: Nonlinear cochlear signal processing. In: *Physiology of the Ear*, 2nd edn., ed. by A. Jahn, J. Santos-Sacchi (Singular, San Diego 2001) pp. 393–442, Chap. 19

3.98 S. Neely: A model of cochlear mechanics with outer hair cell motility, J. Acoust. Soc. Am. **94**, 137–146 (1992)

3.99 J. Ashmore: A fast motile response in guinea-pig outer hair cells: the molecular basis of the cochlear amplifier, J. Physiol. (London) **388**, 323–347 (1987)

3.100 J. Santos-Sacchi: Reversible inhibition of voltage–dependent outer hair cell motility and capacitance, J. Neurosci. **11**(10), 3096–3110 (1991)

3.101 K. Iwasa, R. Chadwick: Elasticity and active force generation of cochlear outer hair cells, J. Acoust. Soc. Am. **92**(6), 3169–3173 (1992)

3.102 I. Russell, P. Legan, V. Lukashkina, A. Lukashkin, R. Goodyear, G. Richardson: Sharpened cochlear tuning in a mouse with a genetically modified tectorial membrane, Nat. Neurosci. **10**, 215–223 (2007)

3.103 D. Sen, J.B. Allen: Functionality of cochlear micromechanics – as elucidated by the upward spread of masking and two tone suppression, Acoustics Australia **34**(1), 43–51 (2006)

3.104 J.B. Allen: Cochlear micromechanics: A physical model of transduction, J. Acoust. Soc. Am. **68**(6), 1660–1670 (1980)

3.105 J.B. Allen: Cochlear micromechanics – A mechanism for transforming mechanical to neural tuning within the cochlea, J. Acoust. Soc. Am. **62**, 930–939 (1977)

3.106 P. Dallos, D.Z. He, X. Lin, I. Sziklai, S. Mehta, B.N. Evans: Acetylcholine, outer hair cell electromotility, and the cochlear amplifier, J. Neurosci. **17**(6), 2212–2226 (1997)

3.107 P. Dallos: Prestin and the electromechanical responses of outer hair cells, ARO-2002 **25**, 189 (2002)

3.108 J.B. Allen: Derecruitment by multiband compression in hearing aids. In: *The Efferent Auditory System*, ed. by C. Berlin (Singular, San Diego 1999) pp. 73–86, Chap. 4 (includes a CDROM video talk by J. B. Allen in MP3 format)

3.109 J.B. Allen, D. Sen: Is tectorial membrane filtering required to explain two tone suppression and the upward spread of masking?. In: *Recent Developments in Auditory Mechanics*, ed. by H. Wada, T. Takasaka, K. Kieda, K. Ohyama, T. Koike (World Scientific, Singapore 1999) pp. 137–143

3.110 W. Sewell: The effects of furosemide on the endocochlear potential and auditory-nerve fiber tuning curves in cats, Hearing Res. **14**, 305–314 (1984)

3.111 J.B. Allen, P.F. Fahey: Nonlinear behavior at threshold determined in the auditory canal on the auditory nerve. In: *Hearing – Physiological Bases and Psychophysics*, ed. by R. Klinke, R. Hartmann (Springer, Berlin, Heidelberg 1983) pp. 128–134

3.112 J.B. Allen, B.L. Lonsbury-Martin: Otoacoustic emissions, J. Acoust. Soc. Am. **93**(1), 568–569 (1993)

3.113 P.F. Fahey, J.B. Allen: Measurement of distortion product phase in the ear canal of cat, J. Acoust. Soc. Am. **102**(5), 2880–2891 (1997)

3.114 R. Diependaal, E. de Boer, M. Viergever: Cochlear power flux as an indicator of mechanical activity, J. Acoust. Soc. Am. **82**, 917–926 (1987)

3.115 G. Zweig: Finding the impedance of the organ of Corti, J. Acoust. Soc. Am. **89**, 1229–1254 (1991)

3.116 E. deBoer, A. Nuttall: The "inverse problem" solved for a three-dimensional model of the cochlear, III Brushing-up the solution method, J. Acoust. Soc. Am. **105**(6), 3410–3420 (1999)

3.117 E. deBoer, A. Nuttall: The mechanical waveform of the basilar membrane, II From data to models – and back, J. Acoust. Soc. Am. **107**(3), 1487–1496 (2000)

3.118 G. Zweig: Finding the impedance of the organ of Corti, J. Acoust. Soc. Am. **89**(3), 1276–1298 (1991)

3.119 E. deBoer, A. Nuttall: The "inverse problem" solved for a three-dimensional model of the cochlea. II Application to experimental data sets, J. Acoust. Soc. Am. **98**(2), 904–910 (1995)

3.120 C. Shera, J. Guinan: Cochlear traveling-wave amplification, suppression, and beamforming probed using noninvasive caliration of intracochlear distortion sources, J. Acoust. Soc. Am. **121**(2), 1003–1016 (2007)

3.121 L.A. Pipes: *Applied Mathematics for Engineers and Physicists* (McGraw-Hill, New York 1958)

3.122 J.B. Allen: Modeling the noise damaged cochlea. In: *The Mechanics and Biophysics of Hearing*, ed. by P. Dallos, C.D. Geisler, J.W. Matthews, M.A. Ruggero, C.R. Steele (Springer, New York 1991) pp. 324–332

3.123 I.J. Russell, G.P. Richardson, A.R. Cody: Mechanosensitivity of mammalian auditory hair cells in vitro, Nature **321**(29), 517–519 (1986)

3.124 E.G. Boring: *History of Psychophysics* (Appleton–Century, New York 1929)

3.125 G. Fechner: Translation of "Elemente der Psychophysik". In: *Elements of Psychophysics*, Vol. I, ed. by H. Adler (Holt Rinehart Winston, New York 1966)

3.126 R. Schlauch, S. Harvey, N. Lanthier: Intensity resolution and loudness in broadband noise, J. Acoust. Soc. Am. **98**(4), 1895–1902 (1995)

3.127 J.B. Allen, S.T. Neely: Modeling the relation between the intensity JND and loudness for pure tones and wide–band noise, J. Acoust. Soc. Am. **102**(6), 3628–3646 (1997)

3.128 J. Zwislocki, H. Jordan: On the relation of intensity JNDs to loudness and neural noise, J. Acoust. Soc. Am. **79**, 772–780 (1986)

3.129 N.F. Viemeister: Psychophysical aspects of auditory intensity coding. In: *Auditory Function*, ed. by G. Edelman, W. Gall, W. Cowan (Wiley, New York 1988) pp. 213–241, Chap. 7

3.130 C. Plack, R. Carlyon: Loudness perception and intensity coding. In: *Hearing, Handbook of Perception and Cognition*, ed. by B. Moore (Academic, San Diego 1995) pp. 123–160, Chap. 4

3.131 J.B. Allen: Harvey Fletcher 1884–1981. In: *The ASA edition of Speech, Hearing in Communication*, ed. by J.B. Allen (Acoust. Soc. Am., Woodbury 1995) pp. 1–34

3.132 D.M. Green, J.A. Swets: *Signal Detection Theory and Psychophysics* (Wiley, New York 1966)

3.133 G.A. Miller: Sensitivity to changes in the intensity of white noise and its relation to masking and loudness, J. Acoust. Soc. Am. **19**, 609–619 (1947)

3.134 W. Hellman, R. Hellman: Intensity discrimination as the driving force for loudness, Application to pure tones in quiet, J. Acoust. Soc. Am. **87**(3), 1255–1271 (1990)

3.135 J. Egan, H. Hake: On the masking pattern of a simple auditory stimulus, J. Acoust. Soc. Am. **22**, 622–630 (1950)

3.136 W. Jesteadt, C. Wier, D. Green: Intensity discrimination as a function of frequency and sensation level, J. Acoust. Soc. Am. **61**(1), 169–177 (1977)

3.137 S. Stevens, H. Davis: *Hearing, Its Psychology and Physiology* (Acoust. Soc. Am., Woodbury 1983)

3.138 W.A. Yost: *Fundamentals of Hearing, An Introduction* (Academic, San Diego, London 1994)

3.139 W. Munson: The growth of auditory sensation, J. Acoust. Soc. Am. **19**, 584–591 (1947)

3.140 W. Siebert: Some implications of the stochastic behavior of primary auditory neurons, Kybernetik **2**, 205–215 (1965)

3.141 D. Raab, I. Goldberg: Auditory intensity discrimination with bursts of reproducible noise, J. Acoust. Soc. Am. **57**(2), 437–447 (1975)

3.142 H.C. Montgomery: Influence of experimental technique on the measurement of differential intensty sensitivity of the ear, J. Acoust. Soc. Am. **7**, 39–43 (1935)

3.143 P.G. Nutting: The complete form of Fechner's law, Bull. Bureau Standards **3**(1), 59–64 (1907)

3.144 H. Fletcher: *Speech and Hearing* (Van Nostrand, New York 1929)

3.145 E.H. Weber: Der Tastsinn und das Gemainfül. In: *Handwörterbuch der Physiologie*, Vol. 3, ed. by R. Wagner (Vieweg, Braunschweig 1988) pp. 481–588, Chap. 7

3.146 J. Johnson, C. Turner, J. Zwislocki, R. Margolis: Just noticeable differences for intensity and their relation to loudness, J. Acoust. Soc. Am. **93**(2), 983–991 (1993)

3.147 B.C.J. Moore: *An Introduction to the Psychology of Hearing*, 2nd edn. (Academic, London, New York 1982)

3.148 W.J. McGill, J.P. Goldberg: A study of the near–miss involving Weber's law and pure tone intensity discrimination, Percept. Psychophys. **4**, 105–109 (1968)

3.149 D. Green: Audition: Psychophysics and perception. In: *Stevens' Handbook of Experimental Psychology*, ed. by R. Atkinson, R. Herrnstein, G. Lindzey, R. Luce (Wiley, New York 1988) pp. 327–376, Chap. 6

3.150 D. Green: Application of detection theory in psychophysics, Proc. IEEE **58**(5), 713–723 (1970)

3.151 S. Stevens: Mathematics, measurement, and psychophysics. In: *Handbook of Experimental Psychology*, ed. by S. Stevens (Wiley, New York 1951) pp. 1–49, Chap. 1

3.152 R. Luce: *Sound and Hearing* (Lawrence Erlbaum, Hilldale 1993)

3.153 S. Stevens: To honor Fechner and repeal his law, Science **133**(3446), 80–86 (1961)

3.154 H. Fletcher: Physical measurements of audition and their bearing on the theory of hearing, Bell System Tech. J. **ii**(4), 145–180 (1923)

3.155 H. Fletcher, J. Steinberg: The dependence of the loudness of a complex sound upon the energy in the various frequency regions of the sound, Phys. Rev. **24**(3), 306–317 (1924)

3.156 J. Steinberg: The loudness of a sound and its physical stimulus, Phys. Rev. **26**, 507 (1925)

3.157 H. De Vries: The quantum character of light and its bearing upon the threshold of vision, the differential sensitivity and the acuity of the eye, Physica **10**, 553–564 (1943)

3.158 W. Siebert: Stimulus transformations in the peripheral auditory system. In: *Recognizing Patterns*, ed. by P. Kolers, M. Eden (MIT Press, Cambridge 1968) pp. 104–133, Chap. 4

3.159 W. Siebert: *Personal communication* (1989)

3.160 W.J. McGill, J.P. Goldberg: Pure–tone intensity discrimination as energy detection, J. Acoust. Soc. Am. **44**, 576–581 (1968)

3.161 J.C. Steinberg, M.B. Gardner: On the auditory significance of the term hearing loss, J. Acoust. Soc. Am. **11**, 270–277 (1940)

3.162 H. Fletcher: A method of calculating hearing loss for speech from an audiogram, J. Acoust. Soc. Am. **22**, 1–5 (1950)

3.163 G. Ekman: Weber's law and related functions, Psychology **47**, 343–352 (1959)

3.164 S. Stevens, H. Davis: *Hearing, Its Psychology and Physiology* (Acoust. Soc. Am., Woodbury 1938)

3.165 W.A. Munson: An experimental determination of the equivalent loudness of pure tones, J. Acoust. Soc. Am. **4**, 7 (1932), Abstract

4. Perception of Speech and Sound

B. Kollmeier, T. Brand, B. Meyer

The transformation of acoustical signals into auditory sensations can be characterized by psychophysical quantities such as loudness, tonality, or perceived pitch. The resolution limits of the auditory system produce spectral and temporal masking phenomena and impose constraints on the perception of amplitude modulations. Binaural hearing (i. e., utilizing the acoustical difference across both ears) employs interaural time and intensity differences to produce localization and binaural unmasking phenomena such as the binaural intelligibility level difference, i. e., the speech reception threshold difference between listening to speech in noise monaurally versus listening with both ears.

The acoustical information available to the listener for perceiving speech even under adverse conditions can be characterized using the articulation index, the speech transmission index, and the speech intelligibility index. They can objectively predict speech reception thresholds as a function of spectral content, signal-to-noise ratio, and preservation of amplitude modulations in the speech waveform that enter the listener's ear. The articulatory or phonetic information available to and received by the listener can be characterized by speech feature sets. Transinformation analysis allows one to detect the relative transmission error connected with each of these speech features. The comparison across man and machine in speech recognition allows one to test hypotheses and models of human speech perception. Conversely, automatic speech recognition may be improved by introducing human signal-processing principles into machine processing algorithms.

Acoustically produced speech is a very special sound to our ears and brain. Humans are able to extract the information contained in a spoken message extremely efficiently even if the speech energy is lower than any competing background sound. Hence, humans are able to communicate acoustically even under adverse listening conditions, e.g., in a cafeteria. The process of understanding speech can be subdivided into two stages. First, an auditory pre-processing stage where the speech sound is transformed into its *internal representation* in the brain and special speech features are extracted (such as, e.g., acoustic energy in a certain frequency channel as a function of time, or instantaneous pitch of a speech sound). This process is assumed to be mainly bottom-up with no special preference for speech sounds as compared to other sounds. In other words, the information contained in any of the speech features can be described quite well by the acoustical contents of the input signal to the ears. In a second step, speech pattern recognition takes place under cognitive control where the internally available speech cues are assembled by our brain to convey the underlying message of the speech sound. This

process is assumed to be top-down and cognitive controlled, and is dependent on training, familiarity, and attention. In the context of this handbook we primarily consider the first step while assuming that the second step operates in a nearly perfect way in normal human listeners, thus ignoring the vast field of cognitive psychology, neuropsychology of speech, and psycholinguistics. Instead, we consider the psychoacoustics of transforming speech and other sounds into its *internal representation*. We will concentrate on the acoustical prerequisites of speech perception by measuring and modeling the speech information contained in a sound entering the ear, and finally the speech features that are presumably used by our brain to recognize speech.

4.1 Basic Psychoacoustic Quantities

The ear converts the temporally and spectrally fluctuating acoustic waveform of incoming speech and sound into a stream of auditory percepts. The most important dimensions of auditory perception are:

- the transformation of sound intensity into subjectively perceived loudness,
- the transformation of major frequency components of the sound into subjectively perceived pitch,
- the transformation of different temporal patterns and rhythms into subjectively perceived fluctuations,
- the transformation of the spectro-temporal contents of acoustic signals into subjectively perceived timbre (which is not independent of the dimensions listed above),
- the transformation of interaural disparities (i. e., differences across both ears) and spectro-temporal contents of acoustical signals into the perceived spatial location and spatial extent of an auditory object.

A basic prerequisite for being able to assign these dimensions to a given sound is the ear's ability to internally separate acoustically superimposed sound sources into different auditory streams or objects.

Psychoacoustics is the scientific discipline that measures and models the relation between physical acoustical quantities (e.g., the intensity of a sinusoidal stimulus specified by sound pressure level, frequency, and duration) and their respective subjective impression (e.g., loudness, pitch, and perceived temporal extent).

4.1.1 Mapping of Intensity into Loudness

The absolute threshold in quiet conditions for a continuous sinusoid is highly dependent on the frequency of the pure tone (Fig. 4.1). It shows highest sensitivity in the frequency region around 1 kHz, which also carries most speech information, and increases for low and high frequencies. The normalized threshold in quiet at 1 kHz averaged over a large number of normal hearing subjects is defined as 0 dB sound pressure level (SPL), which corresponds to 20 μPa. As the level of the sinusoid increases, the perceived loudness increases with approximately a doubling of perceived loudness with a level increase by 10 dB. All combinations of sound pressure levels and frequencies of sinusoid that produce the same loudness as a reference 1 kHz sinusoid of a given level (in dB SPL) are denoted as isophones (Fig. 4.1). Hence, the loudness level (in phon) can only be assigned to a sinusoid and not to a multi-frequency mixture of sounds such as speech. This difference is due to the fact that a broadband sound is perceived as being louder than a narrow-band sound at the same

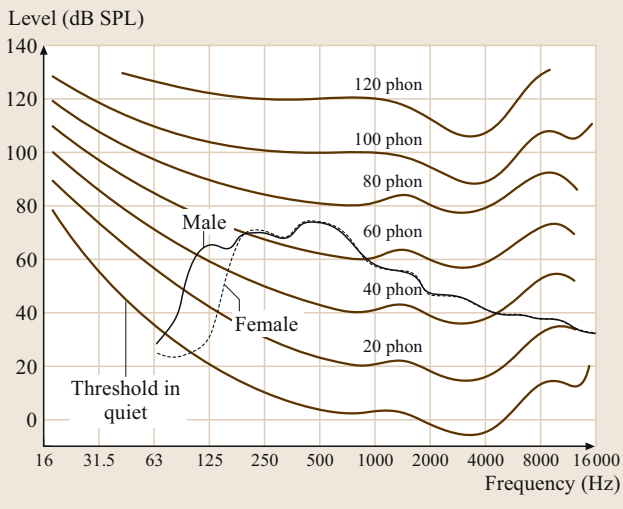


Fig. 4.1 Auditory field described by the threshold in quiet, the isophones, and the uncomfortable listening level of a continuous sinusoid as a function of tone frequency. Also given is an average speech spectrum for male and female speech plotted as a power density

sound pressure level, which puts all of its energy into one *critical band*. In order to express the speech sound pressure level in a way that approximates the human loudness impression, there are several options available:

- Unweighted root-mean-square (RMS) level: the total signal intensity over the audio frequency range is averaged within a certain time window (denoted as *slow*, *fast*, or *impulse*, respectively, for standardized sound level meters, or as the RMS value for a digitized speech signal) and is expressed as dB SPL, i. e., as $10\log(I/I_0)$ where I denotes the signal intensity and I_0 denotes the reference signal intensity at auditory threshold. The usage of the dB scale already takes into account the Weber–Fechner law of psychophysics: roughly, a sound intensity difference of 1 dB can be detected as the just notice-

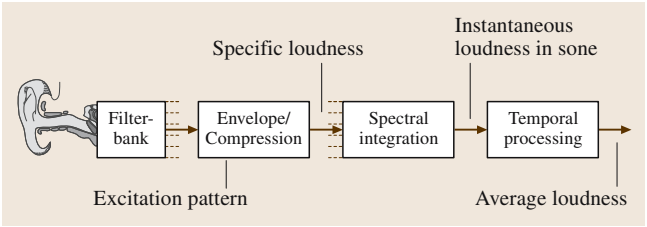


Fig. 4.2 Block diagram of a loudness model

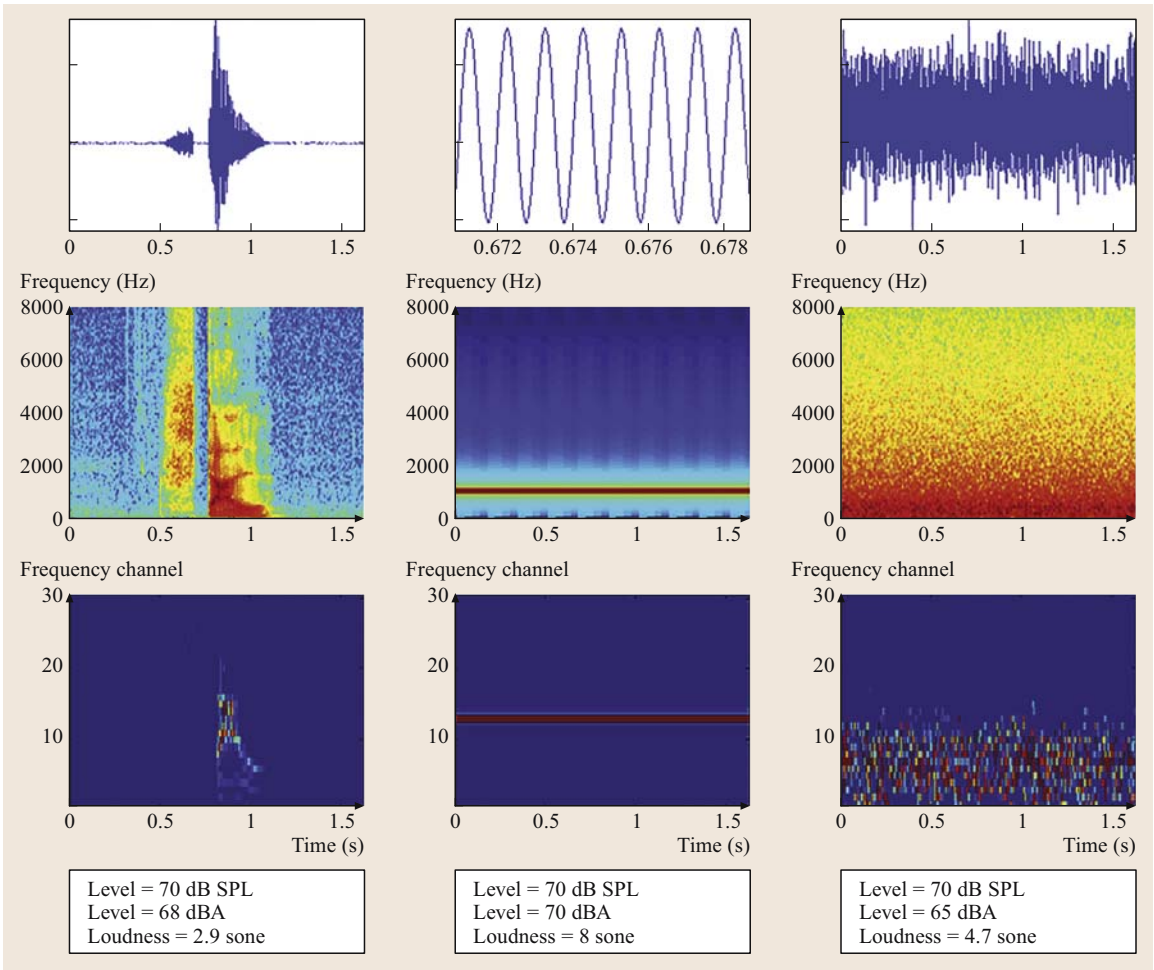


Fig. 4.3 Example plot of a speech sample, 1 kHz sinusoidal tone and a continuous speech-shaped noise sample represented as a waveform, spectrogram, partial loudness pattern and resulting value in dB SPL, dBA and loudness in sone

able intensity difference irrespective of the reference level.

- A-weighted signal level: in order to account for the higher sensitivity of the ear to mid-frequencies at low levels, a spectral weighting function that approximates the isophones between 20 and 30 phon (denoted as A-weighting) is applied to the spectral components of the sound before they are summed up to give the total sound level. The B- and C-weighting curves are available for higher signal levels. Note that the A-, B-, and C-weighted speech levels do not differ too much from the unweighted speech level because the long-term average spectral shape of speech includes most energy in the frequency range close to 1 kHz where the weighting curves coincide. Since none of these definitions include the psychophysical effect of loudness summation across frequency and the temporal integration performed by our ear, a speech sound at a given sound pressure level does not necessarily produce the same perceived loudness as a 1 kHz signal at this level.
- Loudness in sone. A more exact measure of perceived loudness for sounds that differ in frequency contents is given by a loudness calculation scheme based on the sone scale. For a narrow-band sound (like a sinusoid), the loudness in sone is expressed as

$$N[\text{sone}] = (I/I_0)^{\alpha} , \tag{4.1}$$

where I_0 is the reference intensity which is set to 40 dB SPL for a sinusoid at 1 kHz. Since the exponent α amounts to ≈ 0.3 according to *Stevens* and *Zwicker* [4.1], this yields a compression of sound intensity similar to the nonlinear compression in the human ear. For broadband sounds (such as speech), the same compressive power law as given above has to be applied to each *critical* frequency band (see later) before the partial loudness contributions are summed up across frequencies, which results in the total loudness. The detailed loudness calculation scheme (according to ISO 532b) also accounts for the spread of spectral energy of a narrow-band sound into adjacent frequency bands known from cochlear physiology (Sect. 4.1.2). This *leakage* of spectral energy can also be modeled by a bank of appropriately shaped bandpass filters with a limited upper and lower spectral slope (Fig. 4.2).

To account for the time dependency of loudness perception, a temporal integration process is assumed that sums up all intensity belonging to the same auditory object within a period of approximately 200 ms. This roughly models the effect that sounds with a constant sound level are perceived as being louder if their duration increases up to 200 ms while remaining constant in loudness if the duration increases further. For fluctuating sounds with fluctuating instantaneous loudness estimates, the overall loudness impression is dominated by the respective loudness maxima. This can be well represented by considering the 95 percentile loudness (i. e., the loudness value that is exceeded for only 5% of the time) as the *average* loudness value of a sequence of fluctuating sounds [4.1]. For illustration, Fig. 4.3 displays the relation between waveform, spectrogram, partial loudness pattern and resulting value in dB SPL, dBA and loudness in sone for three different sounds.

4.1.2 Pitch

If the frequency of a sinusoid at low frequencies up to 500 Hz is increased, the perceived tone height (or pitch, also denoted as tonality) increases linearly with frequency. At higher frequencies above 1 kHz, however, the perceived pitch increases approximately logarithmically with increasing frequency. The combination of both domains yields the psychophysical mel-scale (Fig. 4.4).

This relation between frequency and subjective frequency perception also represents the mapping of frequencies on the basilar membrane (Sect. 4.2.1 and

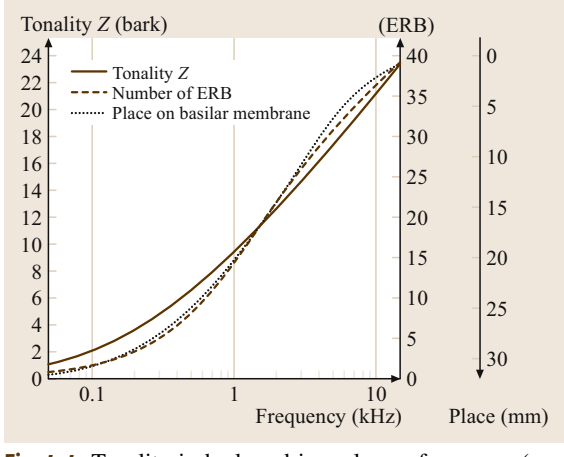


Fig. 4.4 Tonality in bark and in mel over frequency (one bark equals 100 mel). For comparison the (hypothetical) place of maximum excitation on the basilar membrane and the psychoacoustical frequency scale based on equivalent rectangular bandwidth (ERB) is plotted

Fig. 4.4), where frequencies up to approximately 2 kHz occupy half of the basilar membrane and those between 2 kHz and 20 kHz the remaining half. The slope of this function relates to the just noticeable difference (JND) for frequency. The frequency JND is about 3 Hz for frequencies below 500 Hz and about 0.6% for frequencies above 1000 Hz, which is approximately 3 mel. This value amounts to 1/30 of the frequency-dependent bandwidth of the *critical band* which plays a role both in loudness summation (see above) and in the psychophysical effect of spectral masking. All spectral energy that falls into one *critical band* is summed up and *masks* (or disables) the detection of a sinusoidal tone centered within that critical band as long as its level is below this *masked threshold*. According to *Zwicker* et al. [4.2] the auditory critical bandwidth is expressed in bark after the German physicist *Barkhausen* as a function of frequency f_0 (in Hz) as:

$$\begin{aligned} 1\,\text{bark} &= 100\,\text{mel} \\ &\approx 100\,\text{Hz for frequencies below 500 Hz} \\ &\approx 1/5\,f_0 \text{ for frequencies above 500 Hz}\,. \end{aligned} \tag{4.2}$$

The psychophysical frequency scale resulting from integrating the critical bandwidth over frequency is denoted as *bark scale* Z and can be approximated [4.4] by the inverse function of the hyperbolical sinus

$$Z[\text{bark}] = 7 \cdot \text{arcsinh}(f_0/650) \tag{4.3}$$

More-refined measurements of spectral masking performed by *Moore* and *Patterson* [4.5] resulted in the *equivalent rectangular bandwidth* (ERB) as a measure of the psychoacoustical critical bandwidth. It deviates slightly from the bark scale, especially at low frequencies (Fig. 4.4).

It should be noted, however, that the *pitch strength* of a sinusoid decreases steadily as the frequency increases. A much more distinct pitch perception, which is also related to the pitch of musical instruments and the pitch of voiced speech elements, can be perceived for a periodic, broad band sound. For such complex sounds, the term *pitch* should primarily be used to characterize the (perceived) fundamental frequency while *tone height* or *tonality* refers to the psychophysical equivalence of frequency and coincides with pitch only for sinusoids. The perception of pitch results both from temporal cues (i. e., the periodicity information within each critical band) primarily at low frequencies, and from spectral pitch cues, i. e., the regular harmonic structure/line spectrum of a periodic sound primarily dominating at high frequencies. Several theories exist about the perception and relative importance of temporal and spectral pitch cues [4.6]. For the perception of the pitch frequency range of normal speech with fundamental frequencies between approximately 80 Hz and 500 Hz, however, predominantly temporal pitch cues are exploited by our ear. In this range, the pitch JND amounts to approximately 1 Hz, i. e., better resolution occurs for the fundamental frequency of a complex tone than for the audio frequency of a single sinusoid at the fundamental frequency.

4.1.3 Temporal Analysis and Modulation Perception

When perceiving complex sounds such as speech that fluctuate in spectral contents across time, the ear can roughly be modeled as a bank of critical-band wide bandpass filters that transform the speech signal into a number of narrow-band time signals at center frequencies that are equally spaced across the bark scale. Hence, the temporal analysis and resolution within each of these frequency channels is of special importance

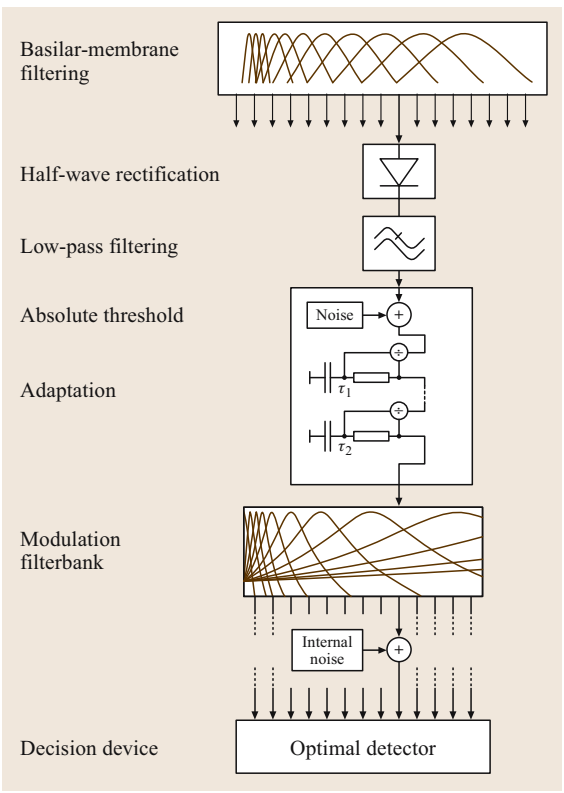


Fig. 4.5 Model of the *effective* signal processing in the auditory system (after [4.3])

for the overall function of our auditory system. For the within-channel analysis, the following phenomena are relevant.

- For center frequencies below ≈ 1000 Hz, the temporal fine structure of the bandpass channel is coded in the auditory nerve and is therefore accessible to the brain. Hence the signal's phase can be exploited during the central processing stages, e.g. for a comparison between different frequency bands to produce a difference in perceived timbre and for a comparison between ears to produce a difference in perceived localization as the phase characteristic is changed. The latter results from extracting the interaural phase difference of a sound signal arriving from a point in space with a certain travel time to either ear (see later).
- For center frequencies above 1 kHz, primarily the envelope of the signal is extracted and analyzed. This makes the ear comparatively phase-deaf above 1 kHz. This envelope extraction is due to the asymmetry between depolarization and hyperpolarization at the synapses between inner hair cells and the auditory nerve as well as due to the temporal integration observed in auditory nerve fibers (Chap. 3). In auditory models this can be modeled to a good approximation by a half-wave rectifier followed by a low-pass filter with a cut-off frequency of ≈ 1 kHz.
- The resulting envelope is subject to a compression and adaptation stage that is required to map the large dynamic range of auditory input signals to the comparatively narrow dynamic range of the nervous system. It is also necessary to set the operation point of the respective further processing stages according to some average value of the current input signal. This compression and adaptation characteristic can either be modeled as a logarithmic compression in combination with the temporal leaky integrator using an *effective* auditory temporal integration window or, alternatively, by a series of nonlinear adaptation loops (Fig. 4.5).

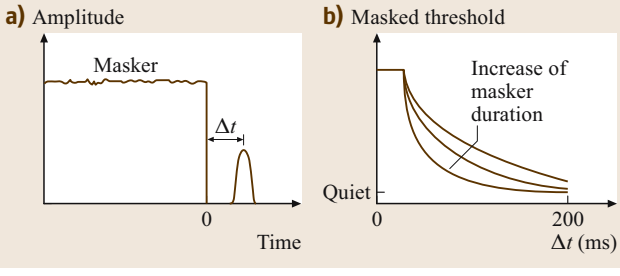


Fig. 4.6a,b Schematic plot of a masked threshold **(a)** of a short probe tone in the presence of (or following) a masking noise burst that extends across a variable amount of time **(b)** (simultaneous and forward masking)

Such an adaptation stage produces the *temporal integration* effect already outlined in Sect. 4.1.1, i. e., all temporal energy belonging to the same acoustical object is summed up within a time window with an effective duration of up to 200 ms. In addition, *temporal masking* is due to this integration or adaptation circuit. A short probe signal (of an intensity higher than the threshold intensity in quiet) will become inaudible in the presence of a *masking* signal if the probe signal is presented either before, during, or after the masker. Hence, the masker extends its masking property both back in time (*backward masking*, extending to approximately 5 ms prior to the onset of the masker), simultaneously with the probe signal (*simultaneous masking*, which becomes less efficient if the masker duration is decreased below 200 ms) and subsequent to the masker (*forward masking*, extending up to 200 ms, Fig. 4.6).

Note that forward masking in speech sounds can prevent detection of soft consonant speech components that are preceded by high-energy vocalic parts of speech.

An important further property of temporal analysis in the auditory system is the perception and analysis of the incoming *temporal envelope fluctuations*. While slow amplitude modulations (modulation frequencies below approximately 4 Hz) are primarily perceived as temporal fluctuations, amplitude modulations between approximately 8 Hz and 16 Hz produce a rolling, R-type roughness percept. Modulations between 16 Hz and approximately 80 Hz are perceived as roughness of a sound. Higher modulation frequencies may be perceived as spectral coloration of the input signal without being resolved in the time domain by the auditory system.

The auditory processing of sounds that differ in their composition of modulation frequencies is best described by the modulation spectrum concept which can be modeled by a modulation filter bank (Fig. 4.5). The separation of different modulation frequencies into separate modulation frequency channels (similar to separating the audio frequencies into different center frequency channels in the inner ear) allows the brain to group together sound elements that are generated from the same sound source even if they interfere with sound elements from a different sound source at the same center frequency. Natural objects are usually characterized by a common

modulation of the emitted frequency components as a function of time. By grouping those sound components that exhibit the same modulation spectrum across different center frequencies, the brain is able to recombine all the sound components of a certain object that are spread out across different audio frequencies. This property of the auditory system is advantageous in performing a figure-background analysis (such as required for the famous *cocktail-party phenomenon*, i. e., a talker can be understood even in the background of a lively party with several interfering voices).

A way of quantitatively measuring the auditory grouping effect is the so-called co-modulation masking release (CMR) depicted in Fig. 4.7.

A probe tone has to be detected against a narrowband, fluctuating noise at the same frequency (*on-frequency masker*). If the adjacent frequency bands are stimulated with uncorrelated noise samples, the threshold of the tone in the modulated noise is comparatively high. However, if the masking noise in the adjacent bands fluctuates with the same amplitude modulations across time (this is usually done by duplicating the on-frequency masker and shifting its center frequency appropriately), the probe tone becomes better audible and a distinct threshold shift occurs. This is called co-modulation masking release. It is mainly due to within-channel cues, i. e., the modulation minima become more distinct with increasing noise bandwidth and hence allow for a better detection of the continuous probe tone at a certain instant of time. It is also due to some across-channel cues and cognitive processing, i. e., the co-modulated components at different frequencies are grouped to form a single auditory object which is distinct from the probe tone. Since speech is usually characterized by a high degree of co-modulation across different frequencies for a single speaker, the co-modulation masking release helps to detect any irregularity which is not co-modulated with the remainder of the speech signal. Such detectable irregularities may reflect, e.g., any speech pathology, a second, faint acoustical object or even speech processing artefacts. The CMR effect is most prominent for amplitude modulation frequencies between approximately 4 Hz and 50 Hz [4.7], which is a region where most of the modulation spectrum energy of speech is located. Hence this effect is very relevant for speech perception.

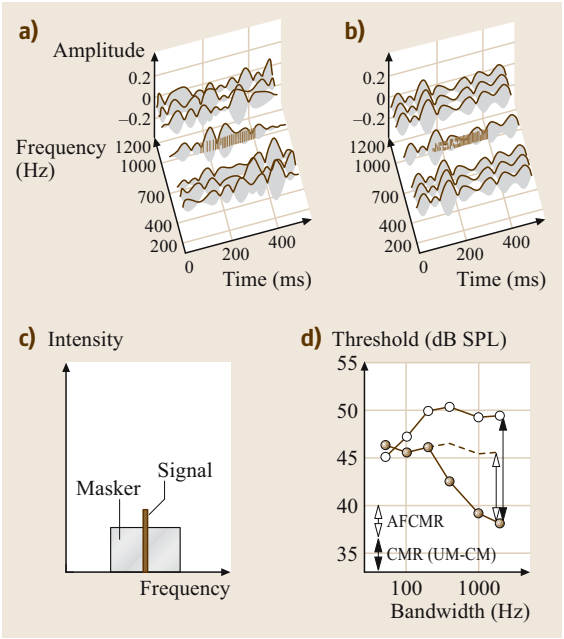


Fig. 4.7a–d Schematic plot of co-modulation masking release (after [4.7]). **(a)** Denotes the temporal-spectral difference in the unmodulated condition (flanking bands are modulated in an uncorrelated way) whereas **(b)** shows the pattern for co-modulated sidebands. In the latter case, the detection of a sinusoid at the on-frequency masking band is facilitated. **(c)**, **(d)** Shows typical psychophysical data for a band-widening experiment with unmodulated (*open symbols*) and co-modulated masker (*filled symbols*). A considerable difference in masked threshold for the sinusoidal signal is observed

4.1.4 Binaural Hearing

Binaural processing, i. e., the central interaction between signal information entering the right and the left ear contributes significantly to

- suppression of subjectively perceived reverberation in closed rooms,
- localization of sound sources in space,
- suppression of *unwanted* sound sources in real acoustical environments.

To perform these tasks, our brain can utilize

- interaural time (or phase) cues, i. e., the central auditory system extracts the travel time difference between the left and right ear,
- interaural intensity difference, i. e., our brain can utilize the head-shadow effect: sound arriving at the ear pointing towards the sound source in space is not attenuated, while the sound arriving at the opposite ear is attenuated,

- spectral changes (coloration) of the sound reaching the inner ear due to interference and scattering effects if the direction of the incoming sound varies (Fig. 4.8).

Normal listeners can localize sound sources with a precision of approximately 1° if sound arrives at the head from the front. This relates to a just noticeable difference (JND) in interaural time difference as small as 10 μs and an interaural level difference JND as low as 1 dB. This remarkable high resolution is due to massive parallel processing at the brain-stem level where the first neural comparison occurs between activation from the right and left ear, respectively.

The binaural performance of our auditory system is extremely challenged in complex acoustical everyday situations characterized by several nonstationary sound sources, reverberation, and a continuous change of the interaural cues due to head movements in space. For perceiving, localizing, and understanding speech in such situations, the following phenomena are relevant:

- Spectral integration of localization cues. Continuous narrow-band signals are hard to localize because their respective interaural time and level difference achieved at the ear level are ambiguous: they can result from any direction within a *cone of confusion*, i. e., a surface that includes all spatial angles centered around the interaural axis that yield the same path difference between right and left ear. For a broadband signal, the comparison across different frequency channels helps to resolve this ambiguity. Also, the onset cues in strongly fluctuating, broadband sounds contain more reliable localization cues than the running cues in the steady-state situation for continuous sounds.
- Precedence effect or the law of the first wavefront. The direct sound (first wavefront) of a sound source hitting the receiver's ears determines the subjective localization percept. Conversely, any subsequent wavefront (that is due to reflections from surrounding structures in a real acoustical environment and hence carries the *wrong* directional information) is not used to create the subjective localization impression. Even though reflections arriving approximately 5–20 ms after the first wavefront are perceivable and their energetic contribution to the total stimulus percept is accessible to the brain, their respective directional information seems to be suppressed. This effect is utilized in some public address loudspeaker systems that deliberately delay the amplified sound in order for the small, unamplified direct sound to reach the listener prior to the amplified sound with the *wrong* directional information.

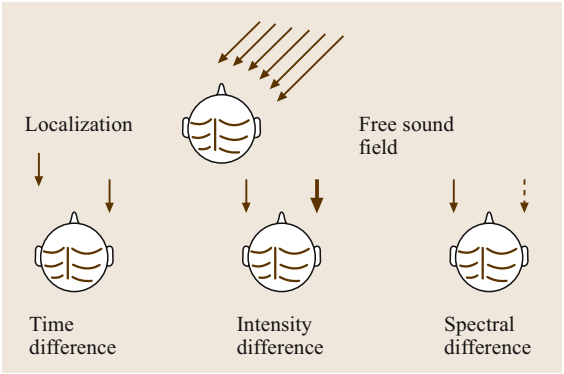


Fig. 4.8 Schematics of interaural cues due to interference and scattering effects that can be utilized by the auditory system

4.1.5 Binaural Noise Suppression

The localization mechanisms described above are not only capable of separating the perceived localization of several simultaneously active acoustical objects. They are also a prerequisite for binaural noise suppression, i. e., an enhancement of the *desired* signal and a suppression of *undesired* parts of the input signals that originate from a different spatial direction. This enhancement is also denoted as binaural release from masking. It can be demonstrated by a tone-in-noise detection experiment where in the reference condition tone and noise are the same at both ears (i. e., exhibit the phase difference 0). The detection threshold can be compared to the threshold using the same noise, but inverting the signal on one side (i. e., a phase difference of π for the signal), which yields a higher detectability. This difference in threshold is denoted as binaural masking level difference and amounts up to 20 dB for short probe tones at frequencies below 1000 Hz.

For speech signals, the binaural unmasking can be measured by comparing the speech reception threshold (i. e., the signal-to-noise ratio required to understand 50% of the presented speech material, see later) for different spatial arrangements of target speech sound and interfering noise: in the reference condition, speech and noise are presented directly in front of the subject, while in the test condition speech comes from the front, but the interfering sound source from the side. The gain in speech reception threshold is called the intelligibil-

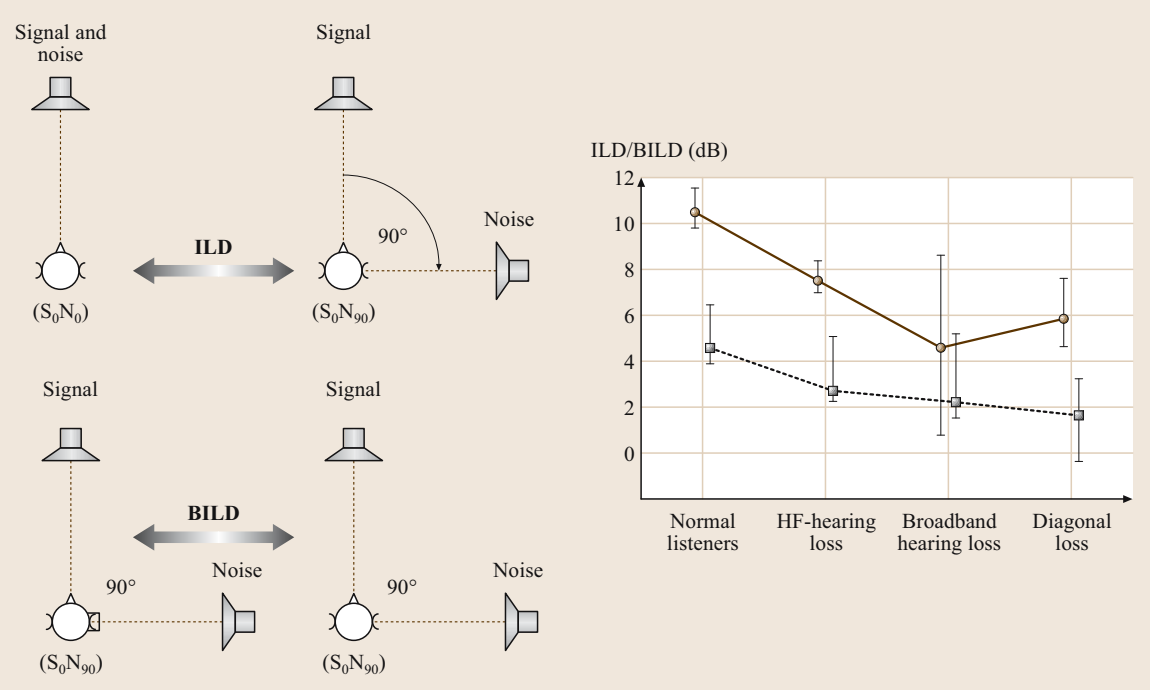


Fig. 4.9 Intelligibility level difference (ILD, *filled circles*) and binaural intelligibility level difference (BILD, *filled squares*) averaged across a group of normal and impaired listeners that differ in the shape of their respective audiogram (high-frequency hearing loss abbreviated as HF-hearing loss). The difference in speech reception threshold across both situations plotted on the left-hand side is plotted as average value and intersubject standard deviation

ity level difference. (The abbreviation ILD is used for this difference, but is also used for interaural level difference.) Intelligibility level difference is due to a monaural effect (i. e., improved signal-to-noise ratio at the ear opposite to the interfering sound source) and a binaural effect. To separate this latter effect, another threshold in the same spatial situation is used where the *worse* ear is plugged and the speech reception threshold is obtained using only the *better* ear, i. e., the ear with the better signal-to-noise ratio. The difference in speech reception threshold (SRT= between the latter two situations (i. e., the difference due to *adding* the *worse* ear) is a purely binaural effect, called the binaural intelligibility level difference (BILD). Figure 4.9 gives an example of the ILD and BILD at an incidence angle for the interfering noise of 90° in an anechoic condition for different groups of listeners that vary in their hearing loss.

A basic model which describes binaural unmasking phenomena quite well for speech signals in complex acoustical environments is a multichannel equalization and cancelation (EC) model such as that depicted in Fig. 4.10 [4.8].

Within each frequency band, an *equalization and cancelation* mechanism [4.9] is used that first delays

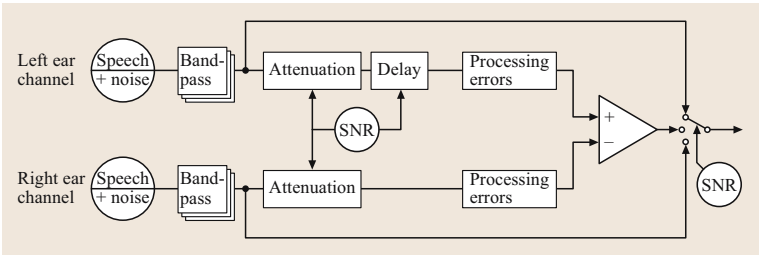


Fig. 4.10 Sketch of a multichannel binaural noise canceling model describing the binaural release from masking for speech in complex environments (after [4.8])

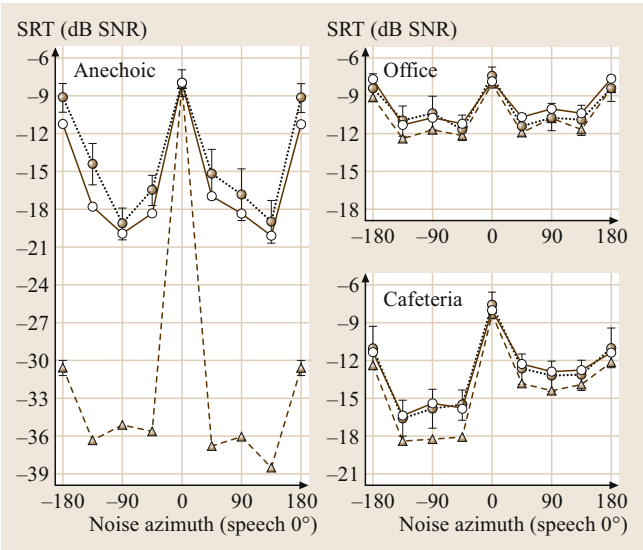


Fig. 4.11 SRT data (*filled symbols*) and predictions for three different acoustical conditions and normal listeners. The *triangles* denote model predictions without introducing appropriate processing errors, whereas the open symbols denote predictions employing internal processing errors, that have been taken from average values in other psychoacoustical tasks (after [4.8])

and amplifies one or both input channels to yield an approximate match (*equalization*) of the composite input signal within each frequency band. In a second, the *cancelation* stage, the signals from both respective sides of the head are (imperfectly) subtracted from each other. Hence, if the masker (after the *equalization* step) is approximately the same in both ears, the cancelation step will eliminate the masker with the exception of some remaining error signal. Conversely, the *desired* signal, which differs in interaural phase and/or intensity relation from the masker, should stay nearly unchanged, yielding an improvement in signal-to-noise ratio. Using an appropriate numerical optimization strategy to fit the respective equalization parameters across frequency, the model depicted in Fig. 4.11 can predict human performance quite well even under acoustically *difficult* situations, such as, e.g., several interfering talkers within a reverberant environment. Note that this model effectively corresponds to an adaptive spatial beam former, i. e., a frequency-dependent optimum linear combination of the two sensor inputs to both ears that yields a directivity optimized to improve the signal-to-noise ratio for a given target direction and interfering background noise. If the model output is used to predict speech intelligibility with an appropriate (monaural) speech intelligibility prediction method [such as, e.g., the speech intelligibility index (SII), see later], the binaural advantage for speech intelligibility in rooms can be predicted quite well (Fig. 4.11 from [4.8]).

Note that in each frequency band only one EC circuit is employed in the model. This reflects the empirical evidence that the brain is only able to cancel out one direction for each frequency band at each instant of time. Hence, the processing strategy adopted will use appropriate compromises for any given real situation.

4.2 Acoustical Information Required for Speech Perception

4.2.1 Speech Intelligibility and Speech Reception Threshold (SRT)

Speech intelligibility (SI) is important for various fields of research, engineering, and diagnostics for quantifying very different phenomena such as the quality of recordings, communication and playback devices, the reverberation of auditoria, characteristics of hearing impairment, benefit using hearing aids, or combinations of these topics. The most useful way to define SI is: *speech intelligibility SI is the proportion of speech items (e.g., syllables, words, or sentences) correctly repeated by (a) listener(s) for a given speech intelligibility test*. This operative definition makes SI directly and quantitatively measurable.

The intelligibility function (Fig. 4.12) describes the listener's speech intelligibility SI as a function of speech level L which may either refer to the sound pressure level (measured in dB) of the speech signal or to the speech-to-noise ratio (SNR) (measured in dB), if the test is performed with interfering noise.

In most cases it is possible to fit the logistic function SI (L) to the empirical data

$$\mathrm{SI}(L)=\frac{1}{A}\left(1+\mathrm{SI}_{\max}\frac{A-1}{1+\exp\left(-\frac{L-L_{\mathrm{mid}}}{s}\right)}\right), \quad (4.4)$$

with L_{mid}: speech level of the midpoint of the intelligibility function; s: slope parameter, the slope at L_{mid} is

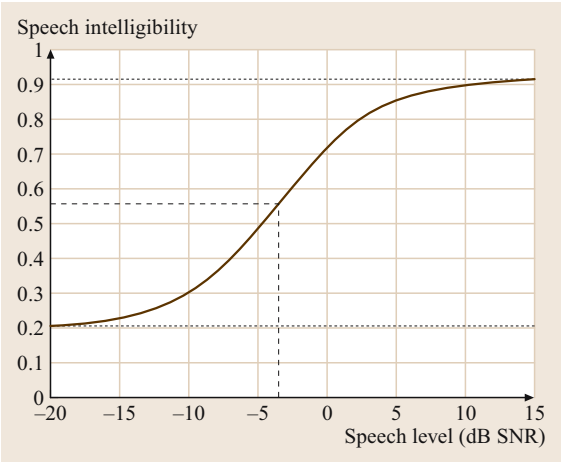


Fig. 4.12 Typical example of SI function (*solid line*) for word intelligibility test (closed response format with five response alternatives). The *dashed line* denotes L_{mid}. The *dotted lines* denote the lower limit $(1/A)$ and the asymptotic maximum $\mathrm{SI}_{\mathrm{asymp}}$ of the SI function. Parameters: $L_{\mathrm{mid}} = -3.5$ dB SNR, $\mathrm{SI}_{\mathrm{max}} = 0.9(\mathrm{SI}_{\mathrm{asymp}} = 0.92)$, $A = 5$, slope $= 0.05$/dB ($s = 3.6$ dB)

given by $\frac{\mathrm{SI}_{\mathrm{max}}(A-1)}{4As}$; $\mathrm{SI}_{\mathrm{max}}$: parameter for maximum intelligibility which can be smaller than 1 in some cases (e.g., distorted speech signals or listeners with hearing impairment). The asymptotic maximum of SI is given by $\mathrm{SI}_{\mathrm{max}} + (1 - \mathrm{SI}_{\mathrm{max}})/A$. A is the number of response alternatives (e.g., $A = 10$ when the listener should respond in a closed response format for instance using digits between '0' and '9'). In SI tests with *open response format*, like word tests without limiting the number of response alternatives, A is assumed to be infinite, that means

$$\mathrm{SI} = \mathrm{SI}_{\mathrm{max}} \frac{1}{1 + \exp\left(-\frac{L - L_{\mathrm{mid}}}{s}\right)} \quad \text{and}$$

$$\text{slope} = \frac{\mathrm{SI}_{\mathrm{max}}}{4s} \,. \tag{4.5}$$

The primary interest of many applications is the speech reception threshold (SRT) which denotes the speech level (measured in dB), which belongs to a given intelligibility (e.g., SI = 0.5 or 0.7).

The accuracy of SI measurements is given by the binomial distribution. Consequently, the standard error SE(SI) of an SI estimate based on n items (e.g., words) is given by

$$\mathrm{SE}(\mathrm{SI}) = \sqrt{\frac{\mathrm{SI}(1 - \mathrm{SI})}{n}} \,. \tag{4.6}$$

A further increase of this standard error is caused by the fact that SI tests consist of several items (e.g., 50 words) which unavoidably differ in SI. Therefore, SI tests should be constructed in a way that the SI of all items is as homogeneous as possible.

To a first approximation, the standard error of the SRT is equal to $\mathrm{SE}(\mathrm{SI}_{\mathrm{SRT}})$ (the standard error of the SI estimate at the SRT) divided by the slope of the intelligibility function at the SRT. Thus

$$\mathrm{SE}(\mathrm{SRT}) = \frac{\mathrm{SE}(\mathrm{SI}_{\mathrm{SRT}})}{\mathrm{slope}_{\mathrm{SRT}}} \,. \tag{4.7}$$

4.2.2 Measurement Methods

Speech Materials

A speech material (i. e., a set of speech items like words or sentences) is suitable for SI tests when certain requirements are fulfilled: the different speech items have to be homogeneous in SI to yield high measurement accuracy and reproducibility in a limited measuring time, and the distribution of phonemes should be representative of the language being studied. Only speech materials that have been optimized properly by a large number of evaluation measurements can fulfill these requirements.

A large number of SI tests using different materials are available for different languages. An overview of American SI tests can be found in *Penrod* [4.10]. There are different formats, i. e., nonsense syllables, single words, and sentences. Sentences best represent a realistic communication situation. Nonsense syllables and words allow assessing of confusion matrices and analyzing transmission of information. Furthermore, the intelligibility functions of most sentence tests [4.11–16] show slopes between 0.15 and 0.25 per dB, which are considerably steeper than the values obtained with nonsense syllables or single-word tests.

Since the standard deviation of SRT estimates is inversely proportional to the slope of the intelligibility function (see Sect. 4.2.1), these sentence tests are better suited for efficient and reliable SRT measurements than single-word tests.

Presentation Modes

Signals can be presented either via loudspeakers (free field condition) or via headphones. The free field condition is more natural. Drawbacks are a larger experimental effort and difficulties in calibration. Especially in spatial speech/noise situations (see Sect. 4.2.3), small movements of the listener's head may influence the result of the SI measurement.

The advantages of presentation via headphones are: very good reproducibility for each individual listener, smaller experimental effort, and spatial speech/noise conditions can easily be realized using virtual acoustics. Drawbacks are the individual calibration is complicated because headphones may produce different sound pressures in different ears. Measurements with hearing aids are not possible.

Adaptive procedures can be used to concentrate presentation levels near the SRT, which yields highest efficiency for SRT estimates. In sentence tests, each word can be scored independently, which allows one to design adaptive procedures which converge more efficiently than adaptive procedures usually used in psychoacoustics [4.17].

4.2.3 Factors Influencing Speech Intelligibility

Measuring Method

The various speech materials mentioned above generate different results. Therefore, only standardized speech materials or speech materials with well known reference intelligibility functions should be used.

Noise and Room Acoustics

Noise and reverberation reduce SI. Therefore, if SI in silence is to be measured, environmental noise and reverberation have to be minimized (e.g., using sound insulated cabins and damping headphones). On the other hand, SI measurements can be used to investigate the influence of different noises and room acoustics on SI, which is important for the so called *cocktail-party phenomenon* (see later).

Cocktail-Party Phenomenon

The human auditory system has very impressive abilities in understanding a target talker even if maskers, i. e., competitive sound sources like different talkers, are present at the same time. An interesting review of research on this so-called *cocktail-party phenomenon* can be found in *Bronkhorst* [4.18]. The SI in these multi-talker conditions is influenced by many masker properties such as sound pressure level, frequency spectrum, amplitude modulations, spatial direction, and the number of maskers. The spatial configuration of target speaker and masker plays a very important role. Binaural hearing (hearing with both ears) produces a very effective release from masking (improvement of the SRT) of up to 12 dB compared to monaural hearing (hearing with one ear) [4.18].

Hearing Impairment

An introduction to SI in clinical audiology can be found in *Penrod* [4.10]. Hearing impairment can lead to an increase of the SRT, a decrease of the maximum reachable intelligibility SI_{asymp} and a flatter slope of the intelligibility function. The most difficult situations for hearing impaired listeners are noisy environments with many interfering sound sources (*cocktail-party situation*). Therefore, SI tests in noise are important diagnostic tools for assessing the daily-life consequences of a hearing impairment and the benefit of a hearing aid. SI plays a very important role for the research on and the fitting of hearing aids.

4.2.4 Prediction Methods

Articulation Index (AI), Speech Intelligibility Index (SII), and Speech Transmission Index (STI)

The most common methods for the prediction of speech intelligibility are the articulation index (AI) [4.19–21] which was renamed the speech intelligibility index (SII) [4.22], and the speech transmission index (STI) [4.23, 24] [Table 4.1]. The strength of these models is the large amount of empirical knowledge they are based on. All of these models assume that speech is coded by several frequency channels that carry independent information. This can be expressed by

$$\mathrm{AI} = \sum_i \mathrm{AI}_i \,, \tag{4.8}$$

with AI denoting the cumulative articulation index of all channels and AI_i denoting the articulation index of the single channels (including a weighting of the respective channel).

AI and SII are derived from the speech signal by calculating the signal to noise ratio SNR in the different frequency channels:

$$\mathrm{AI} = \sum_i \frac{W_i(\mathrm{SNR}_i + 15)}{30} \,, \tag{4.9}$$

with W_i denoting a frequency channel weighting factor and SNR_i denoting the signal-to-noise ratio in channel i. W_i depends on the speech material used and takes into account that high frequencies are more important for the recognition of consonants than for the recognition of meaningful sentences. The main differences between the different versions of AI and SII are the way they include nonlinearities like distortion, masking, and broadening of frequency bands.

The speech transmission index (STI) uses the modulation transfer function instead of the SNR and is especially successful for predicting SI in auditoria and rooms, because it explicitly takes into account the flattening of the information-carrying speech envelopes due to reverberation.

The transformation of AI, SII, or STI to speech intelligibility requires a nonlinear transformation that has to be fitted to empirical data. The transformation depends on the kind of speech material used and is usually steeper at its steepest point for materials with context (e.g., sentences) compared to single words.

Statistical Methods

The assumption of independent information in different frequency channels does not hold in all situations because synergetic as well as redundant interactions between different channels occur. The speech recognition sensitivity model [4.25, 26] takes these interactions into account using statistical decision theory in order to model the linguistic entropy of speech.

Functional Method

These methods are based on relatively rough parameterizations of speech (i. e., long-term frequency spectrum and sometimes modulation transfer function). The method (Table 4.1) proposed by *Holube* and *Kollmeier* [4.27], however, is based on physiological and psychoacoustical data and is a combination of a functional model of the human auditory system (Sect. 4.1) and a simple automatic speech recognition system (Sect. 4.3). A drawback of this approach is that there is still a large gap between recognition rates of humans and automatic speech recognition systems (for a review see [4.28], Sect. 4.3).

Table 4.1 Examples of methods for the prediction of speech intelligibility (SI)

Method	Signal parameters	Comments
Articulation index, AI [4.19]	Levels and frequency spectra of speech and noise, kind of speech material	Macroscopic model that describes the influence of the frequency content of speech on intelligibility
Articulation index, AI [4.20]	Levels and frequency spectra of speech and noise, kind of speech material	More complex than French and Steinberg version, describes more nonlinear effects, seldom used
Articulation index, AI [4.21]	Levels and frequency spectra of speech and noise	Simplified version based on [4.19], not in use anymore
Speech intelligibility index, SII [4.22]	Levels and frequency spectra of speech and noise, kind of speech material, hearing loss	Revision of ANSI S3.5-1969, includes spread of masking, standard speech spectra, relative importance of frequency bands
Speech transmission index, STI [4.24]	Modulation transfer function	Predicts the change of intelligibility caused by a speech transmission system (e.g., an auditorium) based on the modulation transfer function of the system
Speech recognition sensitivity model, SRS [4.25, 26]	Levels and frequency spectra of speech and noise, number of response alternatives	Alternative to SII, handles frequency band interactions and is better suited for unsteady frequency spectra
Holube and *Kollmeier* [4.27]	Speech and noise signals, hearing loss	Microscopic modeling of signal processing of auditory system combined with simple automatic speech recognition

4.3 Speech Feature Perception

The information-theoretic approach to describing speech perception assumes that human speech recognition is based on the combined, parallel recognition of several acoustical cues that are characteristic for certain speech elements. While a *phoneme* represents the smallest unit of speech information, its acoustic realization (denoted as *phone*) can be quite variable in its acoustical properties. Such a phone is produced in order to deliver a number of acoustical speech cues to the listener who should be able to deduce from it the underlying phoneme. Each speech cue represents one feature value of more- or less-complex speech features like *voicing*, *frication*, or *duration*, that are linked to phonetics and to perception. These speech feature values are decoded by the listener independently of each other and are used for recognizing the underlying speech element (such as, e.g., the represented phoneme). Speech perception can therefore be interpreted as reception of certain values of several speech features in parallel and in discrete time steps.

Each phoneme is characterized by a unique combination of the underlying speech feature values. The articulation of words and sentences produces (in the sense of information theory) a discrete stream of information via a number of simultaneously active channels (Fig. 4.13).

The spoken realization of a given phoneme causes a certain speech feature to assume one out of several different possible values. For example, the speech feature *voicing* can assume the value one (i. e., voiced sound) or the value zero (unvoiced speech sound). Each of these features is transmitted via its own, specific transmission channel to the speech recognition system of the listener. The *channel* consists of the acoustical transmission channel to the listener's ear and the subsequent decoding of the signal in the central auditory system of the receiver (which can be hampered by a hearing impairment or a speech pathology). The listener recognizes the actually assumed values of certain speech features and combines these features to yield the recognized phoneme.

If $p(i)$ gives the probability (or relative frequency) that a specific speech feature assumes the value i and $p'(j)$ gives the probability (or relative frequency, respectively) that the receiver receives the feature value j, and $p(i, j)$ gives the joint probability that the value j is recognized if the value i is transmitted, then the so-called transinformation T is defined as

$$T = -\sum_{i=1}^{N}\sum_{j=1}^{N} p(i, j)\log_2\left(\frac{p(i)p'(j)}{p(i, j)}\right) . \qquad (4.10)$$

The transinformation T assumes its maximal value for perfect transmission of the input values to the output values, i. e., if $p(i, j)$ takes the diagonal form or any permutation thereof. T equals 0 if the distribution of received feature values is independent of the distribution of input feature values, i. e., if $p(i, j) = p(i)p'(j)$. The maximum value of T for perfect transmission (i. e., $p(i, j) = p(i) = p'(j)$) equals the amount of information (in bits) included in the distribution of input feature values H, i. e.,

$$H = \sum_{i=1}^{N} p(i)H(i) = -\sum_{i=1}^{N} p(i)\log_2[p(i)] . \qquad (4.11)$$

In order to normalize T to give values between 0 and 1, the so-called transinformation index (TI) is often

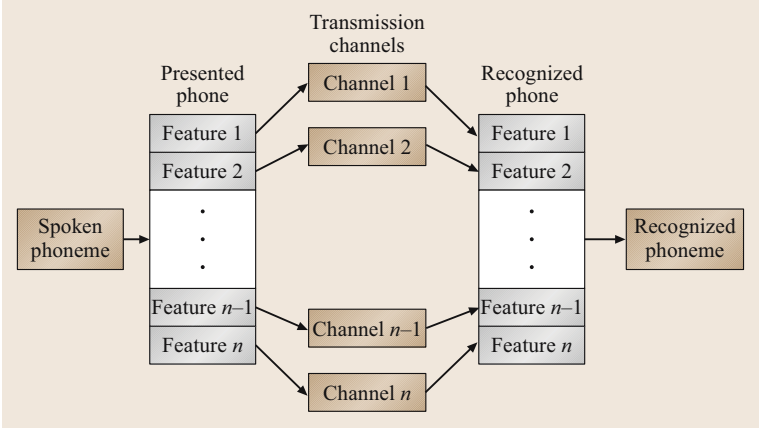


Fig. 4.13 Schematic representation of speech recognition using speech features. Each speech sound is characterized by a combination of speech features that are modeled to be transmitted independently of each other by specialized (noisy) transmission channels

used, i. e.,

$$\mathrm{TI} = T/H \,. \tag{4.12}$$

For speech perception experiments, the distribution $p(i, j)$ can be approximated by a confusion matrix, i. e., a matrix denoting the frequency of a recognized speech element j for all different presented speech elements i. This confusion matrix can be condensed to a confusion matrix of each specific speech feature if for each speech element the corresponding value of the respective feature is assigned. For example, a 20×20 confusion matrix of consonants can be reduced to 2×2 matrix of the feature *voicing* if for each consonant the feature value *voiced* or *unvoiced* is given. The transinformation analysis of this feature-specific confusion matrix therefore allows extracting the transmission of all respective speech features separately and hence can be used to characterize a certain speech information transmission channel. Note however, that such an analysis requires a sufficiently high number of entries in the confusion matrix to appropriately sample $p(i, j)$, which requires a large data set. Also, it is not easy and straightforward to assign appropriate speech features to all of the presented and recognized speech sounds that will allow an adequate analysis of the acoustical deficiencies in the transmission process. From the multitude of different features and feature sets that have been used in the literature to describe both human speech production and speech perception, only a very limited set of the most prominent features can be discussed here ([4.29], for a more-complete coverage).

4.3.1 Formant Features

Vowels are primarily discriminated by their formant structure, i. e., the resonance frequencies of the vocal tract when shaping the vowels. For stationary vowels, the relation between the first and second formant frequencies (F_1 and F_2), respectively, and the perceived vowel is quite well established (Fig. 4.14 and the introduction).

The *classical* theory of vowel perception has the advantage of being linked to the physical process of speech production (i. e., the formant frequencies are closely linked to the position and elevation of the tongue in the vocal tract). However, modern theories of speech perception no longer assume that vowel identification is based solely on the position of the formant frequencies, for the following reasons.

- For short vowels and for vocalic segments of running speech the perceived vowel often differs from the expected spectral shapes, which are based on long, isolated vowels. This indicates that the mapping between formants and perceived vowels shows different boundaries and different regions of overlap depending on the respective speech context.
- The spectral shape of speech in real-life environments varies considerably due to large spectral variations of the room transfer function, and due to the presence of reverberation and background noise. The pure detection and identification of spectral peaks would yield a much less robust perception of vowels than can actually be observed in human listeners.
- Vowel discrimination and speech understanding is even possible under extreme spectral manipulations, such as, e.g., flat spectrum speech [4.30] and slit-filtered speech (i. e., listening to speech through very few spectral slits, [4.31]).

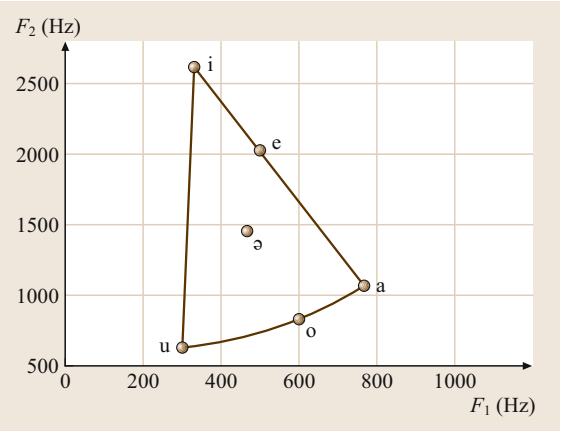


Fig. 4.14 Schematic plot of the perceived vowels as a function of F_1 and F_2 in the vowel triangle (sometimes plotted as a quadrangle) for stationary vowels. Note that the vowel boundaries overlap and that for short vowels and for segments of vowels in real speech these boundaries can vary significantly

These findings indicate that speech perception is at least partially based on temporal cues rather than purely on the detection of spectral peaks or formants. Modern speech perception theories therefore assume that our brain monitors the temporal intensity pattern in each frequency band characterized by a *critical band* filter (Sect. 4.1). By comparing these temporal patterns across a few center frequencies that are not too closely spaced, a reliable estimate of the presented vowel is possible. Such principles are both implemented in state-of-the-art perception models (Sect. 4.1) and in preprocessing/feature extraction strategies for automatic speech recognition systems (Sect. 4.2.3 and the chapters in part E).

4.3.2 Phonetic and Distinctive Feature Sets

The *classical* theory of consonant perception assumes that several phonetically and acoustically defined features are used by the auditory system to decode the underlying, presented consonant. A distinctive feature set combines all binary features that characterize all available consonants in a unique way [4.32]. Both articulatory and (to a lesser degree) acoustical features can be employed to construct such a feature set (see Table 4.2 for an example of a feature set and resulting confusion matrices).

These feature sets where extended and used by *Miller* and *Nicely* [4.33], by *Wang* and *Bilger* [4.34] and subsequently by a large number of researchers to characterize the listeners' ability to discriminate across consonants using, e.g., transinformation analysis (Sect. 4.2.4). Using this approach, the amount of information carried by the specific feature that the receiving side was able to use can be characterized quite well. For example, the confusion matrix listed in Table 4.2 yields a total information transmission index of 0.53 with the features *voicing* assumed to be 0.53 and place 0.46.

However, these phonetic features show only a very weak link to the auditory features actually used by human listeners. From the view point of modern auditory models that assume multichannel temporal energy recording and analysis, most of the phonetic features listed above can be regarded as special prototypes of temporal-spectral patterns that are used by our cognitive system to perform a pattern match between actually presented speech and a stored speech reference database in our brain. Hence, they represent some complex combination of basic auditory perception features that might be defined psychoacoustically or physiologically rather than phonetically.

Table 4.2 Example for a consonant feature set and the construction of confusion matrices for speech-in-noise recognition data with normal-hearing listeners. *Top right panel*: phonetic feature values for eleven consonants. Voicing is a binary feature (with feature values 0 and 1), while manner and place are ternary features. *Middle*: matrix of confusion for consonants, obtained from human listening tests, where noisy speech was presented at an SNR of -10 dB. Matrix element (i, j) denotes how often the consonant in row i was confused with the consonant in column j. *Bottom panels*: confusion matrices for the phonetic features place and voicing, derived from the matrix of confusion for consonants

	Voicing	Manner	Place
p	0	0	0
t	0	0	1
k	0	0	2
b	1	0	0
d	1	0	1
g	1	0	2
s	0	1	1
f	0	1	0
v	1	1	0
n	1	2	2
m	1	2	1

	p	t	k	b	d	g	s	f	v	n	m
p	379	20	131	45	7	31		49	46	4	5
t	3	658	16		33			1	1		
k	42	14	484	10	8	117	1	16	12	6	
b	58	4	51	260	35	88		18	143	16	25
d	5	28	7	21	424	93	1	3	19	49	6
g	11	5	44	43	27	449		9	73	18	7
s		2					702	3			
f	23		3	4			88	556	38		
v	19	7	16	78	22	43	7	51	398	9	30
n	1	5	3	13	51	12		2	23	364	78
m	7	1	4	43	20	25		8	62	95	346

	Anterior	Medail	Posterior
Anterior	2691	335	392
Medail	179	2317	131
Posterior	223	79	1094

	Voiced	Unvoiced
Voiced	3508	375
Unvoiced	367	3191

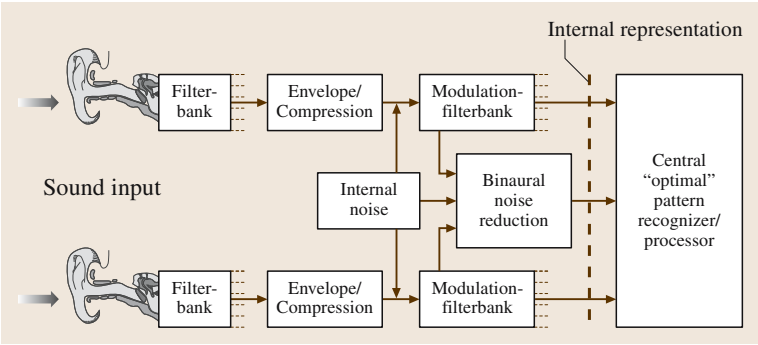


Fig. 4.15 Schematic diagram of a model of the *effective* auditory processing using a front end to transform the incoming speech signal into and internal representation and a subsequent *back end*, ideal recognition stage which is only limited by the internal noise

4.3.3 Internal Representation Approach and Higher-Order Temporal-Spectral Features

The *internal representation* approach of modeling speech reception assumes that the speech signal is transformed by our auditory system with some nonlinear, parallel processing operations into an *internal representation*. This representation is used as the input for a central, cognitive recognition unit which can be assumed to operate as an *ideal observer*, i. e., it performs a pattern match between the incoming internal representation and the multitude of stored internal representations. The accuracy of this recognition process is limited by the *external variability* of the speech items to be recognized, i. e., by their deviation from any of the stored internal templates. It is also limited by the *internal noise* that blurs the received *internal representation* due to neural noise and other physiological and psychological factors. The amount of internal noise can be estimated quite well from psychoacoustical experiments.

Such an internal representation model puts most of the peculiarities and limitations of the speech recognition process into the nonlinear, destructive transformation process from the acoustical speech waveform into its internal representation, assuming that all transformation steps are due to physiological processes that can be characterized completely physiologically or by psychoacoustical means (Fig. 4.15).

Several concepts and models to describe such an internal representation have been developed so far. Some of the basic ideas are as follows.

1. Auditory spectrogram: The basic internal representation assumes that the speech sound is separated into a number of frequency bands (distributed evenly across a psychoacoustically based frequency scale like the bark or ERB scale) and that the compressed frequency-channel-specific intensity is represented

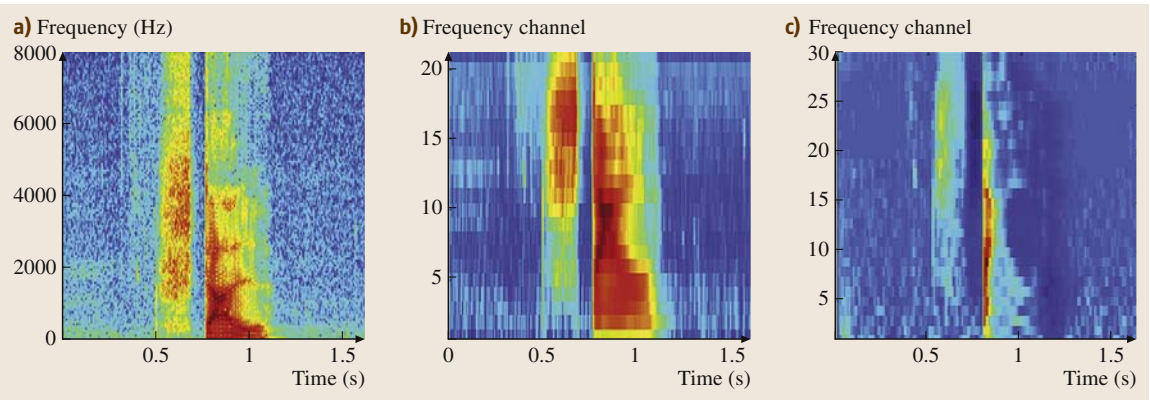


Fig. 4.16a–c Auditory spectrogram representation of the German word *Stall*. It can be represented by a spectrogram **(a)**, a bark spectrogram on a log-loudness scale **(b)** or as a contrast-enhanced version using nonlinear feedback loops (after [4.3, 35]) **(c)**

over time. The compression can either be a logarithmic compression or a loudness-derived power law compression that is also required to represent human intensity resolution and loudness mapping. The temporal representation can also include some temporal contrast enhancement and sluggishness in order to represent forward and backward masking and temporal integration (Sect. 4.1). An example of such a representation is given in Fig. 4.16.

Note that speech intelligibility in noise can be modeled quite well with such an approach [4.27]. In addition, such a transformation into the *internal representation* can be implemented as a robust front end for automatic speech recognition (e.g., [4.35]). Finally, it can be used to predict any perceived deviations of the (coded) speech from the original speech [4.36].

2. Modulation spectrogram: one important property of the internal representation is the temporal analysis within each audio frequency band using the modulation filter bank concept. Temporal envelope fluctuations in each audio frequency channel are spectrally analyzed to yield the modulation spectrum in each frequency band, using either a fixed set of modulation filters (modulation filter bank) or a complete spectral analysis (modulation spectrum). This representation yields the so-called amplitude modulation spectrogram for each instant of time, i. e., a two-dimensional representation of modulation frequency across center audio frequencies (Fig. 4.17).

The physiological motivation for this analysis is the finding of amplitude modulation sensitivity in the auditory brain:adjacent cells are tuned to different modulation frequencies. Their arrangement seems to yield a perpendicular representation of modulation frequencies across center frequencies [4.37]. In addition, psychoacoustical findings of modulation sensitivity can best be described by a set of modulation filter banks [4.3]. The advantage of the modulation spectrogram is that the additional dimension of modulation frequency allows separation of acoustical objects that occupy the same center frequency channel, but are modulated at different rates (considering either the syllabic rate at low modulation frequencies or temporal pitch at higher modulation frequencies). Such a more-refined model of internal representation has been used to predict psychoacoustical effects [4.3] and was also used in automatic speech recognition [4.38].

3. Temporal/spectral ripple or Gabor feature approach: A generalization of the modulation frequency feature detectors in the temporal domain outlined above also considers the spectral analysis of ripples in

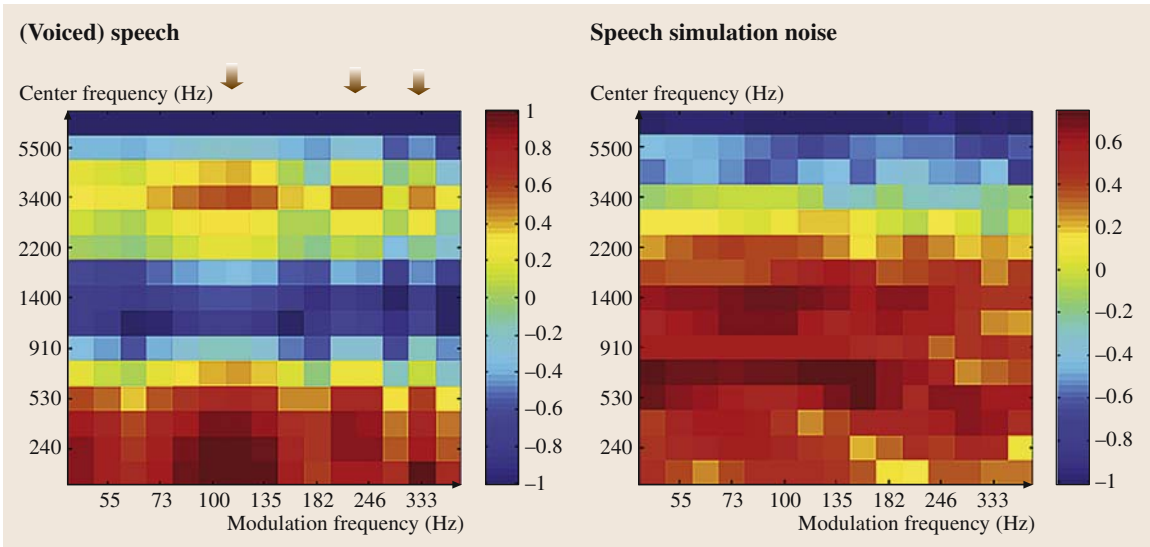


Fig. 4.17 Amplitude modulation spectrogram of a vowel *I* (fundamental frequency approx. 110 Hz) in comparison to a modulation spectrogram of speech-simulating noise. The modulation spectrum in each audio frequency band is displayed as color (or greyscale, respectively) in the two-dimensional plane given by audio center frequency versus modulation frequency

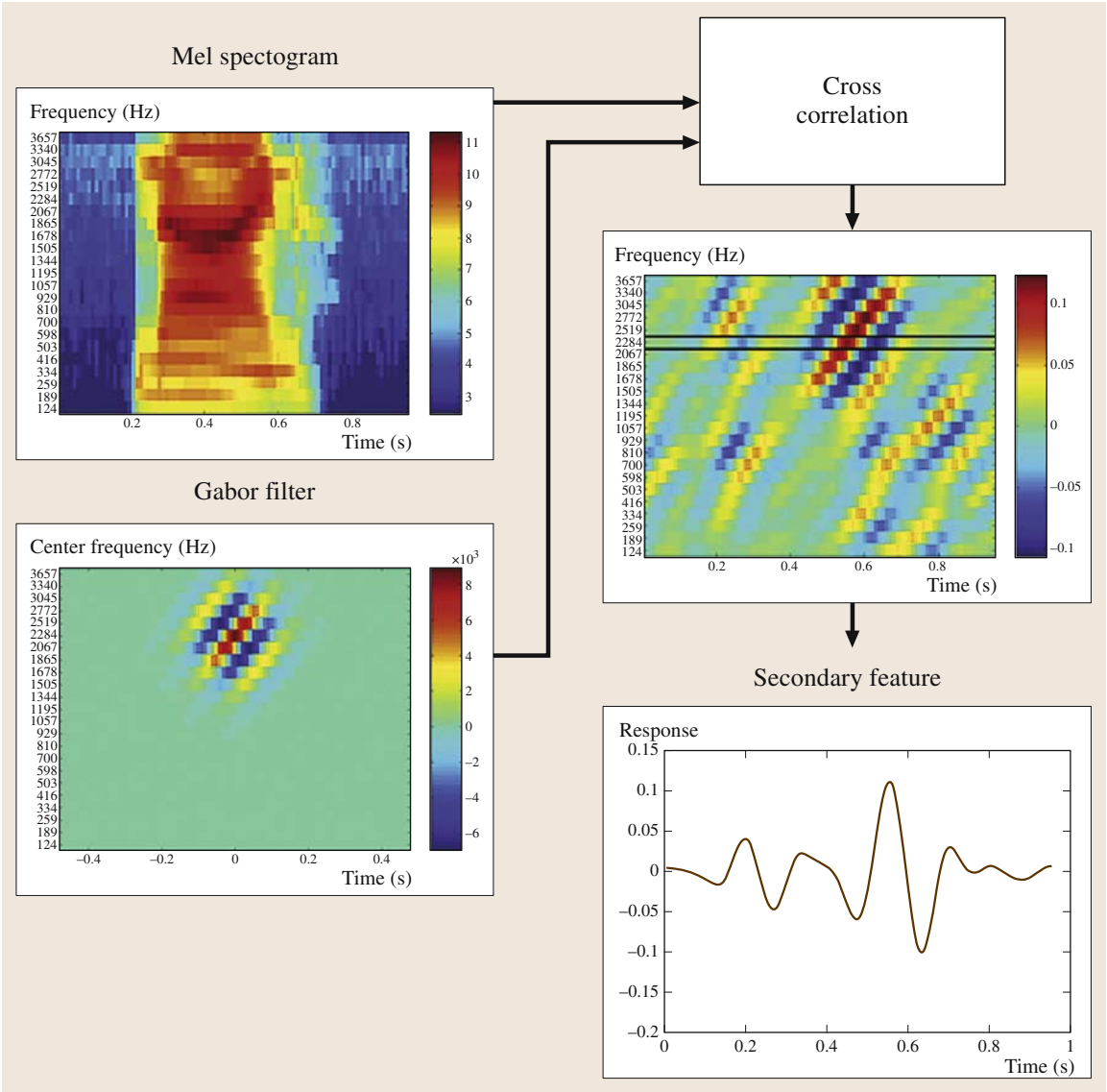


Fig. 4.18 Sample representation of a Gabor feature that detects a certain speech feature. The two-dimensional Gabor feature (*lower left panel*) that extends both in the time and frequency domain is cross-correlated with the mel spectrogram (*upper-left panel*) to yield the temporal and spatial position of a best match (*middle panel*). In each audio frequency band, the time-dependent output of the cross correlation is used as the input feature to an automatic speech recognizer [4.38]

the frequency domain as well as a ripple frequency analysis for combined temporal and spectral modulations. Such a temporal-spectral ripple analysis is motivated by physiological findings of the auditory receptive fields in ferrets [4.39] as well as psychoacoustical findings by *Kaernbach* [4.40] who demonstrated a sensitivity towards combinations of spectral variations and temporal variations. An elegant way to formalize the sensitivity to joint temporal and spectral energy variations is the Gabor feature concept [4.38] that considers features with a limited spectro-temporal extent tuned to a certain combina-

Table 4.3 Recognition rates (in percent correct) for human speech recognition (HSR) at a signal-to-noise ratio (SNR) of −10 dB, compared to automatic speech recognition (ASR) accuracies at several signal-to-noise ratios. The recognition task for ASR and for human listeners was to classify the middle phoneme in simple nonsense words, which were combinations of either consonant–vowel–consonant or vowel–consonant–vowel. The average rates are broken down into consonant and vowel recognition. At +10 dB SNR, ASR reaches an overall performance that is comparable to HSR at −10 dB SNR. If the same SNR of −10 dB is employed for ASR, error rates are almost 50% higher than for HSR

Condition		Average	Consonants	Vowels
HSR	−10 dB	74.5	67.7	80.5
ASR				
	clean	80.4	85.2	76.2
	15 dB	76.1	77.7	74.6
	10 dB	74.6	75.6	73.7
	5 dB	69.8	69.5	70.0
	0 dB	59.2	55.4	62.5
	−5 dB	49.8	41.0	57.5
	−10 dB	28.4	20.8	35.0

tion of temporal modulation frequency and spectral ripple frequency (Figure 4.18).

The advantage of such a second-order receptive field (i. e., the sensitivity to a certain combination of a spectral and a temporal cue) is the ability to detect specific spectro-temporal structures, e.g., formant glides or changes of fundamental frequency. It can also be considered as a generalization of the concepts outlined in this section. Even though this approach has successfully been implemented to improve the robustness of automatic speech recognizers [4.38], it has not yet been used to model human speech perception.

4.3.4 Man–Machine Comparison

Despite enormous technical advances in recent years, automatic speech recognition (ASR) still suffers from a lack of performance compared to human speech recognition (HSR), which prevents this technology from being widely used. Recognition accuracies of machines drop dramatically in acoustically adverse conditions, i. e., in the presence of additive or convolutive noise, which clearly demonstrates the lack of robustness. For complex tasks such as the recognition of spontaneous speech, ASR error rates are often an order of magnitude higher than those of humans [4.28]. If no high-level grammatical information can be exploited (as in a simple phoneme recognition task), the difference in performance gets smaller, but still remains very noticeable. For example, the HSR consonant recognition rate derived from the confusion matrix in Table 4.2 is 67.7%. The ASR score for the very same task (i. e., the same speech signals at an SNR of −10 dB), obtained with a common recognizer is 20.8%, which corresponds to a relative increase of error rates of 144% (Table 4.3).

This large gap underlines that current state-of-the-art ASR technology is by far not as capable as the human auditory system to recognize speech. As a consequence, the fields of ASR and speech perception modeling in humans may benefit from each other. Since the human auditory system results from a long biological evolution process and seems to be optimally adjusted to perform robust speech recognition, ASR may profit from auditory front ends which are based upon physiological findings and incorporate principles of our hearing system. Ideally, the feature matrix extracted from a speech sound which is used to classify the respective speech element should resemble the internal representation of that speech sound in our brain as closely as possible. Since this internal representation can be approximated by an auditory model, such an auditory model seems to be a good preprocessing stage for a speech recognizer [4.35].

On the other hand, models of the signal processing in the human auditory system can be evaluated using ASR, because–under ideal conditions–human speech perception and its model realization as anthropomorphic ASR system should yield a similar recognition performance and error pattern in well-defined acoustical conditions. Thus, modeling human speech perception can benefit from the computational methods developed in ASR.

References

4.1 H. Fastl, E. Zwicker: *Psychoacoustics: Facts and Models* (Springer, Berlin-Heidelberg 2005)

4.2 E. Zwicker, G. Flottorp, S.S. Stevens: Critical bandwidth in loudness summation, J. Acoust. Soc. Am. **29**, 548 (1957)

4.3 T. Dau, B. Kollmeier, A. Kohlrausch: Modeling auditory processing of amplitude modulation. I. Detection and masking with narrow-band carriers, J. Acoust. Soc. Am. **102**, 2892–2905 (1997)

4.4 M.R. Schroeder: *Computer Speech: Recognition, Compression, Synthesis* (Springer, Berlin-Heidelberg 2005)

4.5 B.C.J. Moore, R.D. Patterson: *Auditory Frequency Selectivity* (Plenum, New York 1986)

4.6 A.J. Houtsma: Pitch perception. In: *Handbook of Perception and Cognition: Hearing*, ed. by B.C.J. Moore (Academic, London 1995) pp. 267–295

4.7 J. Verhey, D. Pressnitzer, I.M. Winter: The psychophysics and physiology of co-modulation masking release, Exp. Brain Res. **153**, 405–417 (2003)

4.8 R. Beutelmann, T. Brand: Prediction of speech intelligibility in spatial noise and reverberation for normal-hearing and hearing-impaired listeners J. Acoust. Soc. Am. **120**(1), 33–42 (2006)

4.9 H.S. Colburn, N.I. Durlach: Models of binaural interaction. In: *Handbook of Perception*, Vol. 4 (Academic, New York 1978) pp. 467–518

4.10 J.P. Penrod: Speech threshold and word recognition/ discrimination testing. In: *Handbook of Clinical Audiology, 4th edn*, ed. by J. Katz (Williams and Wilkins, Baltimore 1994) pp. 147–164

4.11 R. Plomp, A. Mimpen: Improving the reliability of testing the speech-reception threshold for sentences, Audiology **18**, 43–52 (1979)

4.12 B. Hagerman: Sentences for testing speech intelligibility in noise, Scand. Audiol. **11**, 79–87 (1982)

4.13 M. Nilsson, S.D. Soli, J.A. Sullivan: Development of the hearing in noise test for the measurement of speech reception thresholds in quiet and in noise, J. Acoust. Soc. Am. **95**(2), 1085–1099 (1994)

4.14 B. Kollmeier, M. Wesselkamp: Development and evaluation of a German sentence test for objective and subjective speech intelligibility assessment, J. Acoust. Soc. Am. **102**, 2412–2421 (1997)

4.15 K. Wagener, V. Kühnel, B. Kollmeier: Entwicklung und Evaluation eines Satztests für die deutsche Sprache I: Design des Oldenburger Satztests (Development and evaluation of a German sentence test I: Design of the Oldenburg sentence test), Zeitschrift für Audiologie **38**, 4–15 (1999)

4.16 K. Wagener, J.L. Josvassen, R. Ardenkjaer: Design, optimization and evaluation of a Danish sentence test in noise, Int. J. Audiol. **42**(1), 10–17 (2003)

4.17 T. Brand, B. Kollmeier: Efficient adaptive procedures for threshold and concurrent slope estimates for psychophysics and speech intelligibility tests, J. Acoust. Soc. Am. **111**(6), 2801–2810 (2002)

4.18 A. Bronkhorst: The cocktail party phenomenon: a review of research on speech intelligibility in multiple-talker conditions, Acustica **86**, 117–128 (2000)

4.19 N.R. French, J.C. Steinberg: Factors governing the intelligibility of speech sounds, J. Acoust. Soc. Am. **19**, 90–119 (1947)

4.20 H. Fletcher, R.H. Galt: The perception of speech and its relation to telephony, J. Acoust. Soc. Am. **22**, 89–151 (1950)

4.21 ANSI: *Methods for the calculation of the articulation index, ANSI S3.5-1969* (American National Standards Institute, New York 1969)

4.22 ANSI: *Methods for calculation of the speech intelligibility index, ANSI S3.5-1997* (American National Standards Institute, New York 1997)

4.23 T. Houtgast, H.J.M. Steeneken: A review of the MTF concept in room acoustics and its use for estimating speech intelligibility in auditoria, J. Acoust. Soc. Am. **77**, 1069–1077 (1985)

4.24 IEC: Sound system equipment – Part 16: Objective rating of speech intelligibility by speech transmission index. INTERNATIONAL STANDARD 60268-16 Second edition 1998-03 (1998)

4.25 H. Müsch, S. Buus: Using statistical decision theory to predict speech intelligibility. I. Model structure, J. Acoust. Soc. Am. **109**, 2896–2909 (2001)

4.26 H. Müsch, S. Buus: Using statistical decision theory to predict speech intelligibility, II. Measurement and prediction of consonant-discrimination performance J. Acoust. Soc. Am. **109**, 2910–2920 (2001)

4.27 I. Holube, B. Kollmeier: Speech intelligibility prediction in hearing-impaired listeners based on a psychoacoustically motivated perception model, J. Acoust. Soc. Am. **100**, 1703–1716 (1996)

4.28 R. Lippmann: Speech recognition by machines and humans, Speech Commun. **22**, 1–15 (1997)

4.29 S. Greenberg, W.A. Ainsworth, A.N. Popper: Speech Processing in the Auditory System. In: *Handbook of Auditory research*, Vol. 18, ed. by R.R. Fay (Springer, New York 2004)

4.30 M.R. Schroeder, H.W. Strube: Flat spectrum speech, J. Acoust. Soc. Am. **79**, 1580–1583 (1986)

4.31 R.V. Shannon, F.G. Zeng, V. Kamth, J. Wygonsky, M. Ekelid: Speech recognition with primarily temporal cues, Science **270**, 303–304 (1995)

4.32 R. Jakobson, C.G.M. Fant, M. Halle: *Preliminaries to speech analysis: the distinctive features and their correlates* (MIT Press, Cambridge 1963)

4.33 G.A. Miller, P.E. Nicely: An analysis of perceptual confusions among some english consonants, J. Acoust. Soc. Am. **27**, 338–352 (1955)

4.34 M.D. Wang, R.C. Bilger: Consonant confusions in noise: a study of perceptual features, J. Acoust. Soc. Am. **54**, 1248–1266 (1973)
4.35 J. Tchorz, B. Kollmeier: A model of auditory perception as front end for automatic speech recognition, J. Acoust. Soc. Am. **106**(4), 2040–2050 (1999)
4.36 M. Hansen, B. Kollmeier: Objective modeling of speech quality with a psychoacoustically validated auditory model, J. Audio Eng. Soc. **48**(5), 395–408 (2000)
4.37 C.E. Schreiner, G. Langner: Periodicity coding in the inferior colliculus of the cat II. Topographical organization, J. Neurophys. **60**, 1823–1840 (1988)
4.38 M. Kleinschmidt: Methods for capturing spectro-temporal modulations in automatic speech recognition, Acustica united with Acta Acustica **88**(3), 416–422 (2002)
4.39 D.A. Depireux, J.Z. Simon, D.J. Klein, S.A. Shamma: Spectro-temporal response field characterization with dynamic ripples in ferret primary auditory cortex, J. Neurophysiol. **85**(3), 1220–1234 (2001)
4.40 C. Kaernbach: The Memory of Noise, Exp. Psychol. **1**(4), 240–248 (2004)

2. Physiological Processes of Speech Production

K. Honda

Speech sound is a wave of air that originates from complex actions of the human body, supported by three functional units: generation of air pressure, regulation of vibration, and control of resonators. The lung air pressure for speech results from functions of the respiratory system during a prolonged phase of expiration after a short inhalation. Vibrations of air for voiced sounds are introduced by the vocal folds in the larynx; they are controlled by a set of laryngeal muscles and airflow from the lungs. The oscillation of the vocal folds converts the expiratory air into intermittent airflow pulses that result in a buzzing sound. The narrow constrictions of the airway along the tract above the larynx also generate transient source sounds; their pressure gives rise to an airstream with turbulence or burst sounds. The resonators are formed in the upper respiratory tract by the pharyngeal, oral, and nasal cavities. These cavities act as resonance chambers to transform the laryngeal buzz or turbulence sounds into the sounds with special linguistic functions. The main articulators are the tongue, lower jaw, lips, and velum. They generate patterned movements to alter the resonance characteristics of the supra-laryngeal airway. In this chapter, contemporary views on phonatory and articulatory mechanisms are summarized to illustrate the physiological processes of speech production, with brief notes on their observation techniques.

2.1 Overview of Speech Apparatus

The speech production apparatus is a part of the motor system for respiration and alimentation. The form of the system can be characterized, when compared with those of other primates, by several unique features, such as small red lips, flat face, compact teeth, short oral cavity with a round tongue, and long pharynx with a low larynx position. The functions of the system are also uniquely advanced by the developed brain with the language areas, direct neural connections from the cortex to motor nuclei, and dense neural supply to each muscle. Independent control over phonation and articulation is a human-specific ability. These morphological and neural changes along human evolution reorganized the original functions of each component into an integrated motor system for speech communication.

The speech apparatus is divided into the organs of phonation (voice production) and articulation (settings of the speech organs). The phonatory organs (lungs and larynx) create voice source sounds by setting the driving air pressure in the lungs and parameters for vocal fold vibration at the larynx. The two organs together adjust the pitch, loudness, and quality of the voice, and further generate prosodic patterns of speech. The articulatory organs give resonances or modulations to the voice source and generate additional sounds for some consonants. They consist of the lower jaw, tongue,

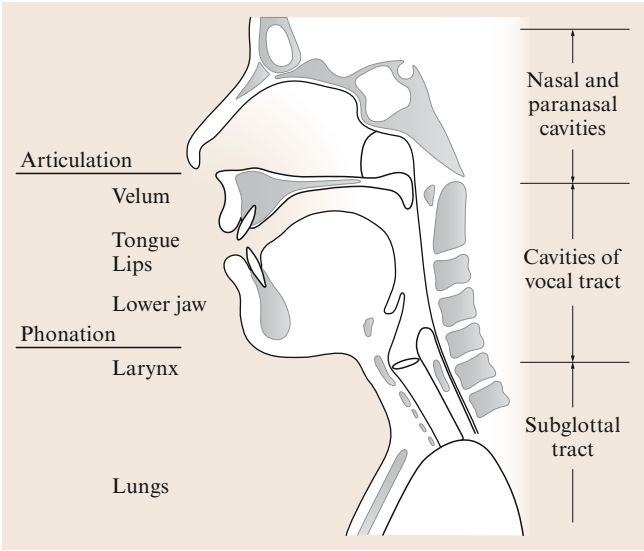


Fig. 2.1 Sketch of a speech production system. Physiological processes of speech production are realized by combined sequential actions of the speech organs for phonation and articulation. These activities result in sound propagation phenomena at the three levels: subglottal cavities, cavities of the vocal tract, and nasal and paranasal cavities

lips, and the velum. The larynx also takes a part in the articulation of voiced/voiceless distinctions. The tongue and lower lip attach to the lower jaw, while the velum is loosely combined with other articulators. The constrictor muscles of the pharynx and larynx also participate in articulation as well as in voice quality control. The phonatory and articulatory systems influence each other mutually, while changing the vocal tract shape for producing vowels and consonants. Figure 2.1 shows a schematic drawing of the speech production system.

2.2 Voice Production Mechanisms

Generation of voice source requires adequate configuration of the airflow from the lungs and vocal fold parameters for oscillation. The sources for voiced sounds are the airflow pulses generated at the larynx, while those for some consonants (i. e., stops and fricatives) are airflow noises made at a narrow constriction in the vocal tract. The expiratory and inspiratory muscles together regulate relatively constant pressure during speech. The laryngeal muscles adjust the onset/offset, amplitude, and frequency of vocal fold vibration.

2.2.1 Regulation of Respiration

The respiratory system is divided into two segments: the conduction airways for ventilation between the atmosphere and the lungs, and the respiratory tissue of the lungs for gas exchange. Ventilation (i. e., expiration and inhalation) is carried out by movements of the thorax, diaphragm, and abdomen. These movements involve actions of respiratory muscles and elastic recoil forces of the system. During quiet breathing, the lungs expand to inhale air by the actions of inspiratory muscles (diaphragm, external intercostal, etc.), and expel air by the elastic recoil force of the lung tissue, diaphragm, and cavities of the thorax and abdomen. In effort expiration, the expiratory muscles (internal intercostals, abdominal muscles, etc.) come into action. The inspiratory and expiratory muscles work alternately, making the thorax expand and contract during deep breathing.

During speech production, the respiratory pattern changes to a longer expiratory phase with a shorter inspiratory phase during quiet breathing. Figure 2.2 shows a conventional view of the respiratory pattern during

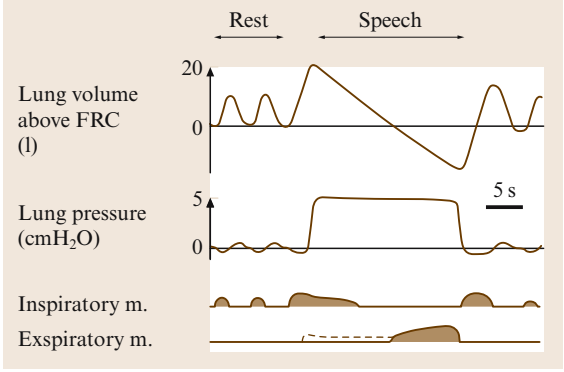


Fig. 2.2 Respiratory pattern during speech. Top two curves show the changes in the volume and pressure in the lungs. The bottom two curves show schematic activity patterns of the inspiratory and expiratory muscles (after [2.1]). The dashed line for the expiratory muscles indicates their predicted activity for expiration

speech [2.1]. The thorax is expanded by inspiration prior to initiation of speech, and then compressed by elastic recoil force by the tissues of the respiratory system to the level of the functional residual capacity (FRC). The lung pressure during speech is kept nearly constant except for the tendency of utterance initial rise and final lowering. In natural speech, stress and emphasis add local pressure increases. The constant lung pressure is due to the actions of the inspiratory muscles to prevent excessive airflow and maintain the long expiratory phase. As speech continues, the lung volume decreases gradually below the level of FRC, and the lung pressure is then maintained by the actions of the expiratory muscles that actively expel air from the lung. It has been argued whether the initiation of speech involves only the elastic recoil forces of the thorax to generate expiratory airflow. Indeed, a few studies have suggested that not only the thoracic system but also the abdominal system assists the regulation of expiration during speech [2.2, 3], as shown by the dashed line in Fig. 2.2. Thus, the contemporary view of speech respiration emphasizes that expiration of air during speech is not a passive process but a controlled one with co-activation of the inspiratory and expiratory muscles.

2.2.2 Structure of the Larynx

The larynx is a small cervical organ located at the top of the trachea making a junction to the pharyngeal cavity: it primarily functions to prevent foreign material from entering the lungs. The larynx contains several rigid structures such as the cricoid, thyroid, arytenoid, epiglottic, and other smaller cartilages. Figure 2.3a shows the arrangement of the major cartilages and the hyoid bone. The cricoid cartilage is ring-shaped and supports the lumen of the laryngeal cavity. It offers two bilateral articulations to the thyroid and arytenoid cartilages at the cricothyroid and cricoarytenoid joints, respectively. The thyroid cartilage is a shield-like structure that offers attachments to the vocal folds and the vestibular folds. The arytenoid cartilages are bilateral tetrahedral cartilages that change in location and orientation between phonation and respiration. The whole larynx is mechanically suspended from the hyoid bone by muscles and ligaments.

The gap between the free edges of the vocal folds is called the *glottis*. The space is divided into two portions by the vocal processes of the arytenoid cartilages: the membranous portion in front (essential for vibration) and cartilaginous portion in back (essential for respiration). The glottis changes its form in various ways during speech: it narrows by adduction and widens by abduction of the vocal folds. Figure 2.3b shows that this movement is carried out by the actions of the intrinsic laryngeal muscles that attach to the arytenoid cartilages. These muscles are functionally divided into the adductor and abductor muscles. The adductor muscles include the thyroarytenoid muscles, lateral cricoarytenoid, and arytenoid muscles, and the abductor muscle is the posterior cricoarytenoid muscle. The glottis also changes in length according to the length of the vocal folds, which takes place mainly at the membranous portion. The length of the glottis shows a large developmental sexual variation. The membranous length on average is 10 mm in adult females and 16 mm in adult males, while the cartilaginous length is about 3 mm for both [2.4].

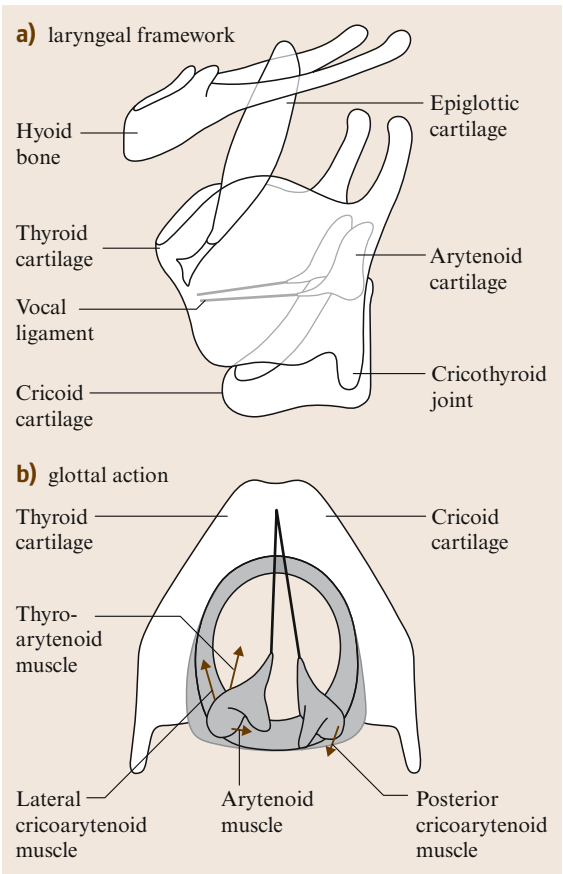


Fig. 2.3a,b Laryngeal framework and internal structures. **(a)** Oblique view of the laryngeal framework, which includes the hyoid bone and four major cartilages. **(b)** Adduction (*left*) and abduction (*right*) of the glottis and the effects of the intrinsic laryngeal muscles

2.2.3 Vocal Fold and its Oscillation

The larynx includes several structures such as the subglottic dome, vocal folds, ventricles, vestibular folds, epiglottis, and aryepiglottic folds, as shown in Fig. 2.4a. The vocal folds run anteroposteriorly from the vocal processes of the arytenoid cartilages to the internal surface of the thyroid cartilage. The vocal fold tissue consists of the thyroarytenoid muscle, vocal ligament, lamina propria, and mucous membrane. They form a special layer structure that yields to aerodynamic forces to oscillate, which is often described as the *body-cover* structure [2.5].

During voiced speech sounds, the vocal folds are set into vibration by pressurized air passing through the membranous portion of the narrowed glottis. The glottal airflow thus generated induces wave-like motion of the vocal fold membrane, which appears to propagate from the bottom to the top of the vocal fold edges. When this oscillatory motion builds up, the vocal fold membranes on either side come into contact with each other, resulting in repetitive closing and opening of the glottis. Figure 2.4b shows that vocal fold vibration repeats four phases within a cycle: the closed phase, opening phase, open phase, and closing phase. The conditions that determine vocal fold vibration are the stiffness and mass of the vocal folds, the width of the glottis, and the pressure difference across the glottis.

The aerodynamic parameters that regulate vocal fold vibration are the transglottal pressure difference and glottal airflow. The former coincides with the measure of subglottal pressure during mid and low vowels, which is about 5–10 cm H_2O in comfortable loudness and pitch (1 cm H_2O = 0.98 hPa). The latter also coincides with the average measure of oral

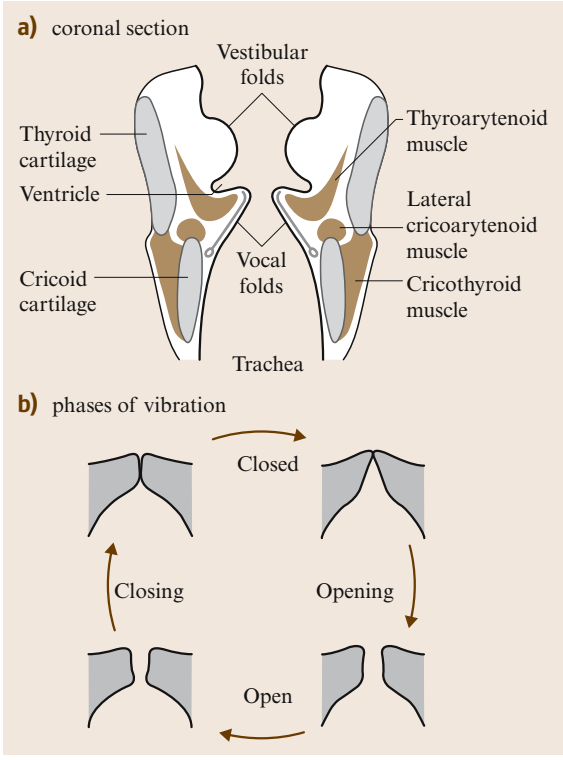


Fig. 2.4a,b Vocal folds and their vibration pattern. **(a)** Coronal section of the larynx, showing the tissues of the vocal and vestibular (false) folds. The cavity of the larynx includes supraglottic and subglottic regions. **(b)** Vocal-fold vibration pattern and glottal shapes in open phases. As the vocal-fold edge deforms in a glottal cycle, the glottis follows four phases: closed, opening, open and closing

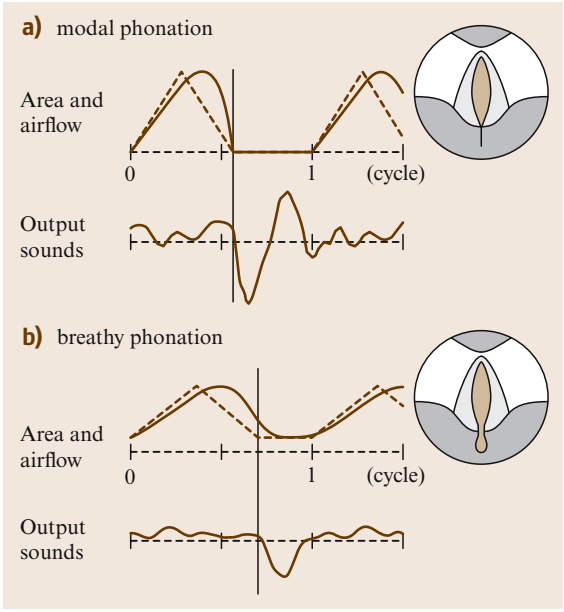


Fig. 2.5a,b Changes in glottal area and airflow in relation to output sounds during 1.5 glottal cycles from glottal opening, with glottal shapes at peak opening (in the circles). **(a)** In modal phonation with complete glottal closure in the closed phase, glottal closure causes abrupt shut-off of glottal airflow and strong excitation of the air in the vocal tract during the closed phase. **(b)** In breathy phonation, the glottal closure is incomplete, and the airflow wave includes a DC component, which results in weak excitation of the tract

airflow during vowel production, which is roughly 0.1–0.2 l/s. These values show a large individual variation: the pressure range is 4.2–9.6 cm H_2O in males and 4.4–7.6 cm H_2O in females, while the airflow rate ranges between 0.1–0.3 l/s in males and 0.09–0.2 l/s in females [2.6].

Figure 2.5 shows schematically the relationship between the glottal cycle and volumic airflow change in normal and breathy phonation. The airflow varies within each glottal cycle, reflecting the cyclic variation of the glottal area and subglottal pressure. The glottal area curve roughly shows a triangular pattern, while the airflow curve shows a skew of the peak to the right due to the inertia of the air mass within the glottis [2.7]. The closure of the glottis causes a discontinuous decrease of the glottal airflow to zero, which contributes the main source of vocal tract excitation, as shown in Fig. 2.5a. When the glottal closure is more abrupt, the output sounds are more intense with richer harmonic components [2.8]. When the glottal closure is incomplete in soft and breathy voices or the cartilaginous portion of the glottis is open to show the *glottal chink*, the airflow includes a direct-current (DC) component and exhibits a gradual decrease of airflow, which results in a more sinusoidal waveform and a lower intensity of the output sounds, as shown in Fig. 2.5b.

Laryngeal control of the oscillatory patterns of the vocal folds is one of the major factors in voice quality control. In sharp voice, the open phase of the glottal cycle becomes shorter, while in soft voice, the open phase becomes longer. The ratio of the open phase within a glottal cycle is called the *open quotient* (OQ), and the ratio of the closing slope to the opening slope in the glottal cycle is called the *speed quotient* (SQ). These two parameters determine the slope of the spectral envelope. When the open phase is longer (high OQ) with a longer closing phase (low SQ), the glottal airflow becomes more sinusoidal, with weak harmonic components. Contrarily, when the open phase is shorter (low OQ), glottal airflow builds up to pulsating waves with rich harmonics. In modal voice, all the vocal fold layers are involved in vibration, and the membranous glottis is completely closed during the closed phase of each cycle. In falsetto, only the edges of the vocal folds vibrate, glottal closure becomes incomplete, and harmonic components reduce remarkably.

The oscillation of the vocal folds during natural speech is quasiperiodic, and cycle-to-cycle variation are observed in speech waveforms as two types of measures: *jitter* (frequency perturbation) and *shimmer* (amplitude perturbation). These irregularities appear to arise from combinations of biomechanical (vocal fold asymmetry), neurogenic (involuntary activities of laryngeal muscles), and aerodynamic (fluctuations of airflow and subglottal pressure) factors. In sustained phonation of normal voice, the jitter is about 1% in frequency, and the shimmer is about 6% in amplitude.

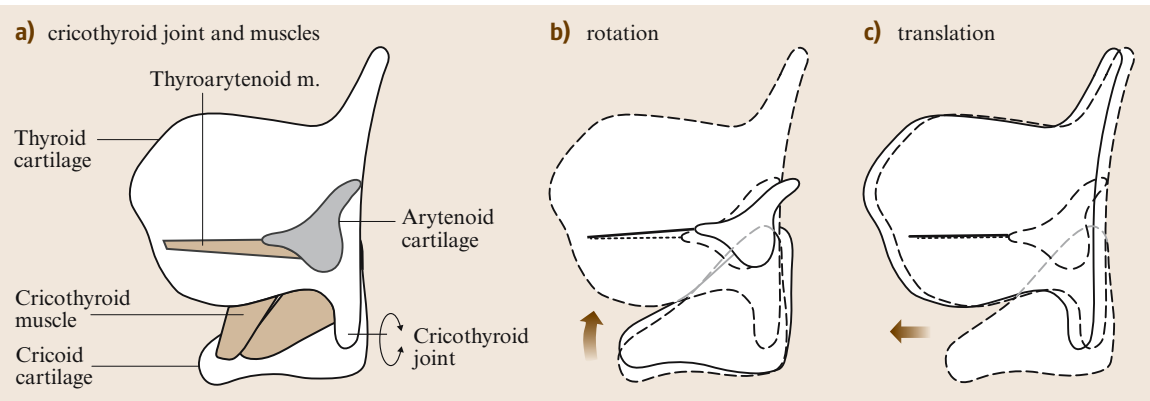


Fig. 2.6a–c Cricothyroid joint and F_0 regulation mechanism. **(a)** The cricothyroid joint is locally controlled by the thyroarytenoid and two parts of the cricothyroid muscles: Pars recta (anterior) and pars obliqua (posterior). As F_0 rises, the thyroid cartilage advances and cricoid cartilage rotates to the direction to stretch the vocal folds, which leads to the increases in the stiffness of vocal fold tissue and in the natural resonance frequency of the vocal folds. **(b)** Rotation of the cricothyroid joint is caused mainly by the action of the pars recta to raise the cricoid arch. **(c)** Translation of the joint is produced mainly by the pars obliqua

2.2.4 Regulation of Fundamental Frequency (F_0)

The fundamental frequency (F_0) of voice is the lowest harmonic component in voiced sounds, which conforms to the natural frequency of vocal fold vibration. F_0 changes depending on two factors: regulation of the length of the vocal folds and adjustment of aerodynamic factors that satisfy the conditions necessary for vocal fold vibration. In high F_0, the vocal folds become thinner and longer; while in low F_0, the vocal folds become shorter and thicker. As the vocal folds are stretched by separating their two attachments (the anterior commissure and vocal processes), the mass per unit length of the vocal fold tissue is reduced while the stiffness of the tissue layer involved in vibration increases. Thus, the mass is smaller and the stiffness is greater for higher F_0 than lower F_0, and it follows that the characteristic frequency of vibrating tissue increases for higher F_0. The length of the vocal folds is adjusted by relative movement of the cricoid and thyroid cartilages. Its natural length is a determinant factor of individual difference in F_0. The possible range of F_0 in adult speakers is about 80–400 Hz in males, and about 120–800 Hz in females.

The thyroid and cricoid cartilages are articulated at the cricothyroid joint. Any external forces applied to this joint cause rotation and translation (sliding) of the joint, which alters the length of the vocal folds. It is well known that the two joint actions are brought about by the contraction of the cricothyroid muscle to approximate the two cartilages at their front edges. Figure 2.6 shows two possible actions of the cricothyroid muscle on the joint: rotation by the pars recta and translation of the pars obliqua [2.9]. Questions still remain as to whether each part of the cricothyroid conducts pure actions of rotation or translation, and as to which part is more responsible for determining F_0.

The extrinsic laryngeal muscles can also apply external forces to this joint as a supplementary mechanism for regulating F_0 [2.10]. The most well known among the activities of the extrinsic muscles in this regulation is the transient action of the sternohyoid muscle observed as F_0 falls. Since this muscle pulls down the hyoid bone to lower the entire larynx, larynx lowering has long been thought to play a certain role in F_0 lowering. Figure 2.7 shows a possible mechanism of F_0 lowering by vertical larynx movement revealed by magnetic resonance imaging (MRI). As the cricoid cartilage descends along the anterior surface of the cervical spine, the cartilage rotates in a direction that shortens the vocal folds because the cervical spine shows anterior convexity at the level of the cricoid cartilage [2.11].

Aerodynamic conditions are an additional factor that alters F_0, as seen in the local rises of the subglottal pressure during speech at stress or emphasis. The increase of the subglottal air pressure results in a larger airflow rate and a wider opening of the glottis, which causes greater deformation of the vocal folds with larger average tissue stiffness. The rate of F_0 increase due to the subglottal pressure is reported to be about 2–5 Hz/cmH_2O when the chest cavity is compressed externally, and is observed to be 5–15 Hz/cmH_2O, when

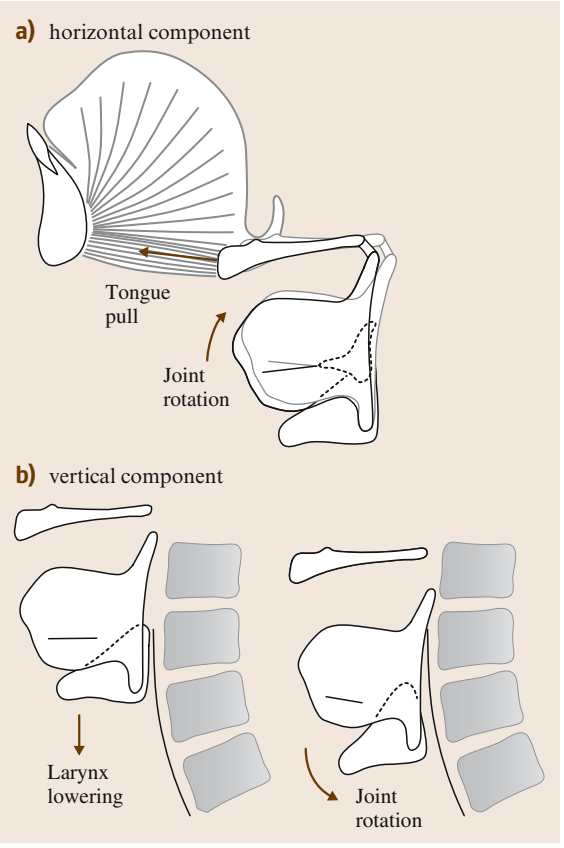


Fig. 2.7a,b Extrinsic control of F_0. Actions of the cricothyroid joint are determined not only by the cricothyroid muscle but also by other laryngeal muscles. Any external forces applied to the joint can activate the actions of the joint. **(a)** In F_0 raising, advancement of the hyoid bone possibly apply a force to rotate the thyroid cartilage. **(b)** In F_0 lowering, the cricoid cartilage rotates as its posterior plate descends along the anterior convexity of the cervical spine

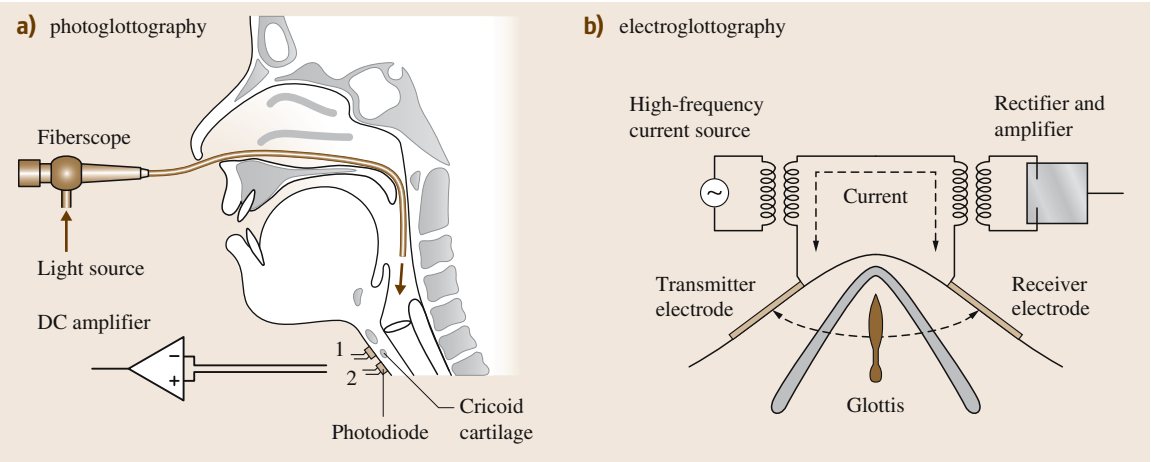


Fig. 2.8a,b Glottographic methods. **(a)** PGG with fiberscopy uses a photodetector attached near the cricothyroid cartilage in two locations: one attachment for measuring vibrations, and two attachment for glottal gestures. **(b)** EGG uses a pair of electrodes on the skin above the thyroid lamina to form a induction circuit to record electrical currents passed through the vocal-fold edges

it is measured between the beginning and end of speech utterances.

2.2.5 Methods for Measuring Voice Production

Speech production mechanisms arise from the functions of the internal organs of the human body that are mostly invisible. Therefore, better understanding of speech production processes relies on the development of observation techniques. The lung functions in speech can be assessed by the tools for aerodynamic measurements, while examination of the larynx functions during speech requires special techniques for imaging and signal recording.

Monitoring Respiratory Functions

Respiratory functions during speech are examined by recording aerodynamic measurements of lung volume, airflow, and pressure. Changes in lung volume are monitored with several types of plethysmography (e.g., whole-body, induction, and magnetic). The airflow from the mouth is measured with pneumotachography using a mask with pressure probes (differential-pressure anemometry) or thermal probes (hot-wire anemometry). Measurements of the subglottal pressure require a tracheal puncture of a needle with a pressure sensor or a thin catheter-type pressure transducer inserted from the nostril to the trachea via the cartilaginous part of the glottis.

Laryngeal Endoscopy

Imaging of the vocal folds during speech has been conducted with a combination of an endoscope and video camera. A solid-type endoscope is capable of observing vocal fold vibration with stroboscopic or real-time digital imaging techniques during sustained phonation. The flexible endoscope is beneficial for video recording of glottal movements during speech with a fiber optic bundle inserted into the pharynx through the nostril via the velopharyngeal port. Recently, an electronic type of flexible endoscope with a built-in image sensor has become available.

Glottography

Glottography is a technique to monitor vocal fold vibration as a waveform. Figure 2.8 shows two types of glottographic techniques. Photoglottography (PGG) detects light intensity modulated by the glottis using an optical sensor. The sensor is placed on the neck and a flexible endoscope is used as a light source. The signal from the sensor corresponds to the glottal aperture size, reflecting vocal fold vibration and glottal adduction–abduction movement. Electroglottography (EGG) records the contact of the left and right vocal fold edges during vibration. High-frequency current is applied to a pair of surface electrodes placed on the skin above the thyroid lamina, which detect a varying induction current that corresponds to the change in vocal fold contact area.

2.3 Articulatory Mechanisms

Speech articulation is the most complex motor activity in humans, producing concatenations of phonemes into syllables and syllables into words using movements of the speech organs. These articulatory processes are conducted within a phrase of a single expiratory phase with continuous changes of vocal fold vibration, which is one of the human-specific characteristics of sound production mechanisms.

2.3.1 Articulatory Organs

Articulatory organs are composed of the rigid organ of the lower jaw and soft-tissue organs of the tongue, lips, and velum, as illustrated in Fig. 2.9. These organs together alter the resonance of the vocal tract in various ways and generate sound sources for consonants in the vocal tract. The tongue is the most important articulatory organ, and changes the gross configuration of the vocal tract. Deformation of the whole tongue determines vowel quality and produces palatal, velar, and pharyngeal consonants. Movements of the tongue apex and blade contribute to the differentiation of dental and alveolar consonants and the realization of retroflex consonants. The lips deform the open end of the vocal tract by various types of gestures, assisting the production of vowels and labial consonants. Actions of these soft-tissue organs are essentially based on contractions of the muscles within these organs, and their mechanism is often compared with the *muscular hydrostat*. Since the tongue and lips have attachments to the lower jaw, they are interlocked with the jaw to open the mouth. The velum controls opening and closing of the velopharyngeal port, and allows distinction between nasal and oral sounds. Additionally, the constrictor muscles of the pharynx adjust the lateral width of the pharyngeal cavity, and their actions also assist articulation for vowels and back consonants.

Upper Jaw

The upper jaw, or the maxilla with the upper teeth, is the structure fixed to the skull, forming the palatal dome on the arch of the alveolar process with the teeth. It forms a fixed wall of the vocal tract and does not belong to the articulatory organs: yet it is a critical structure for speech articulation because it provides the frame of reference for many articulatory gestures. The structures of the upper jaw offer the location for contact or approximation by many parts of the tongue such as the apex, blade, and dorsum. The phonetics literature describes the place of articulation as classified according to the locations of lingual approximation along the upper jaw for dental, alveolar, and palatal consonants. The hard palate is covered by the thick mucoperiosteum, which has several transverse lines of mucosal folds called the *palatine rugae*.

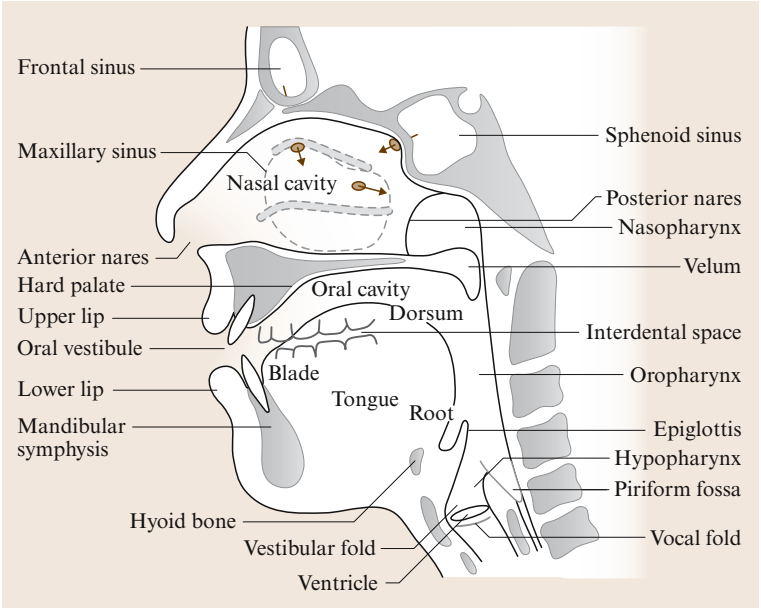


Fig. 2.9 Illustration of the articulatory system with names of articulators and cavities

Lower Jaw

The lower jaw, or the mandible with the lower teeth, is the largest rigid motor organ among the speech production apparatus. Its volume is about 100 cm^3. As well as playing the major role in opening and closing the mouth, it provides attachments for many speech muscles and supports the tongue, lips, and hyoid bone.

Figure 2.10 shows the action of the jaw and the muscles used in speech articulation. The mandible articulates with the temporal bone at the temporomandibular joint (TMJ) and brings about jaw opening–closing actions by rotation and translation. The muscles that control jaw movements are generally called the masticatory muscles. The jaw opening muscles are the digastric and lateral pterygoid muscles. The strap muscles, such as the geniohyoid and sternohyoid, also assist jaw opening. The jaw closing muscles include the masseter, temporalis, and medial pterygoid muscles. While the larger muscles play major roles in biting and chewing, comparatively small muscles are used for speech articulation. The medial pterygoid is mainly used for jaw closing in articulation, and the elastic recoil force of the connective tissues surrounding the mandible is another factor for closing the jaw from its open position.

Tongue

The tongue is an organ of complex musculature [2.12]. It consists of a round body occupying its main mass and a short blade with an apex. Its volume is approximately 100 cm^3, including the muscles in the tongue floor. The tongue body moves in the oral cavity by variously deforming its voluminous mass, while the tongue blade alters its shape and changes the angle of the tongue apex. Deformation of the tongue tissue is caused by contractions of the extrinsic and intrinsic tongue muscles, which are illustrated schematically in Fig. 2.11.

The extrinsic tongue muscles are those that arise outside of the tongue and end within the tongue tissue. This group includes four muscles, the genioglossus, hyoglossus, styloglossus, and palatoglossus muscles, although the former three muscles are thought to be involved in the articulation of the tongue. The palatoglossus muscle participates in the lowering of the velum as discussed later.

The genioglossus is the largest and strongest muscle in the tongue. It begins from the posterior aspect of the mandibular symphysis and runs along the midline of the tongue. Morphologically, it belongs to the triangular muscle, and its contraction effects differ across portions of the muscle. Therefore, the genioglossus is divided functionally into the anterior, middle, and posterior bundles. The anterior and middle bundles run midsagittally, and their contraction makes the midline groove of the tongue for the production of front vowels. The anterior bundle often makes a shallow notch on the tongue surface called the *lingual fossa* and assists elevation of the tongue apex. The middle bundle runs obliquely, and advances the tongue body for front vowels. The posterior bundle of the genioglossus runs midsagittally and

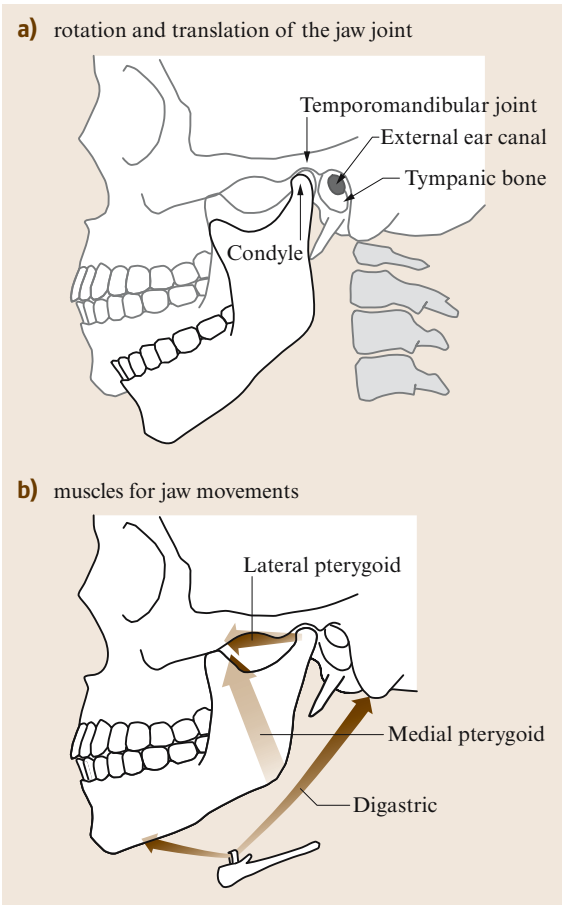


Fig. 2.10a,b Actions of the temporomandibular joint and muscles for jaw opening and closing. **(a)** The lower jaw opens by rotation and translation of the mandible at the temporomandibular joint. Jaw translation is needed for wide opening of the jaw because jaw rotation is limited by the narrow space between the condyle and tympanic bone. **(b)** Jaw opening in speech depends on the actions of the digastric and medial pterygoid muscles with support of the strap muscles. Jaw closing is carried out by the contraction of the lateral pterygoid muscle and elastic recoil forces of the tissues surrounding the jaw

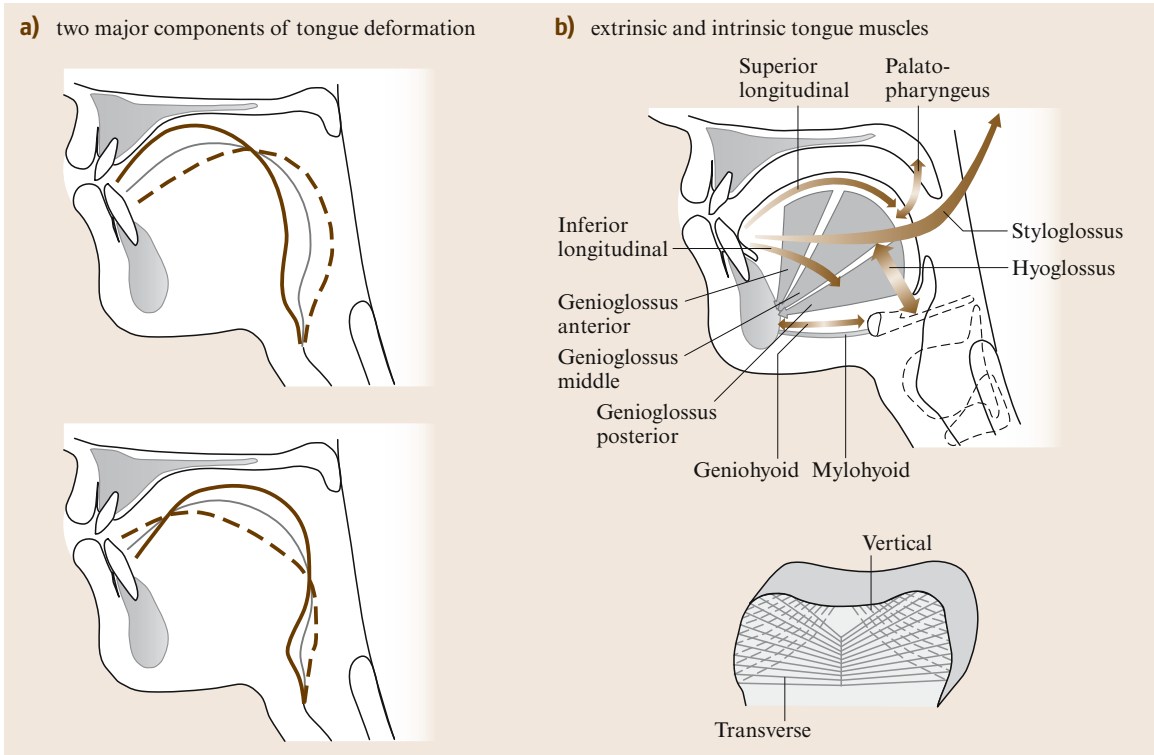


Fig. 2.11a,b Actions of the tongue and its musculature. **(a)** Major components of tongue deformation are high-front vs.low-back (*top*) and high back versus low front (*bottom*) motions, (after [2.14]). **(b)** Lateral view (*top*) shows the extrinsic and intrinsic muscles of the tongue with two tongue floor muscles. Coronal section (*bottom*) shows additional intrinsic muscles

also spreads laterally, reaching a wide area of the tongue root. This bundle draws the tongue root forward and elevates the upper surface of the tongue for high vowels and anterior types of oral consonants. The hyoglossus is a bilateral thin-sheet muscle, which arises from the hyoid bone, runs upward along the sides of the tongue, and ends in the tongue tissue, intermingling with the styloglossus. Its contraction lowers the tongue dorsum and pushes the tongue root backward for the production of low vowels. The styloglossus is a bilateral long muscle originating from the styloid process on the skull base, running obliquely to enter the back sides of the tongue. Within the tongue, it runs forward to reach the apex of the tongue, while branching downward to the hyoid bone and medially toward the midline. Although the extra-lingual bundle of the styloglossus runs obliquely, it pulls the tongue body straight back at the insertion point because the bundle is surrounded by fatty and muscular tissues. The shortening of the intra-lingual bundle draws the tongue apex backward and causes an upward bunching of the tongue body [2.13]. Each of the extrinsic tongue muscles has two functions: drawing of the relevant attachment point toward the origin, and deforming the tongue tissue in the orthogonal orientation. The resulting deformation of the tongue can be explained by two antagonistic pairs of extrinsic muscles: posterior genioglossus versus styloglossus, and anterior genioglossus versus hyoglossus. The muscle arrangement appears to be suitable for tongue body movements in the vertical and horizontal dimensions.

The intrinsic tongue muscle is a group of muscles that have both their origin and termination within the tongue tissue. They include four bilateral muscles: the superior longitudinal, inferior longitudinal, transverse, and vertical muscles. The superior and inferior longitudinal muscles operate on the tongue blade to produce vertical and horizontal movements of the tongue tip. The transverse and vertical muscles together compress the tongue tissue medially to change the cross-sectional shape of the tongue.

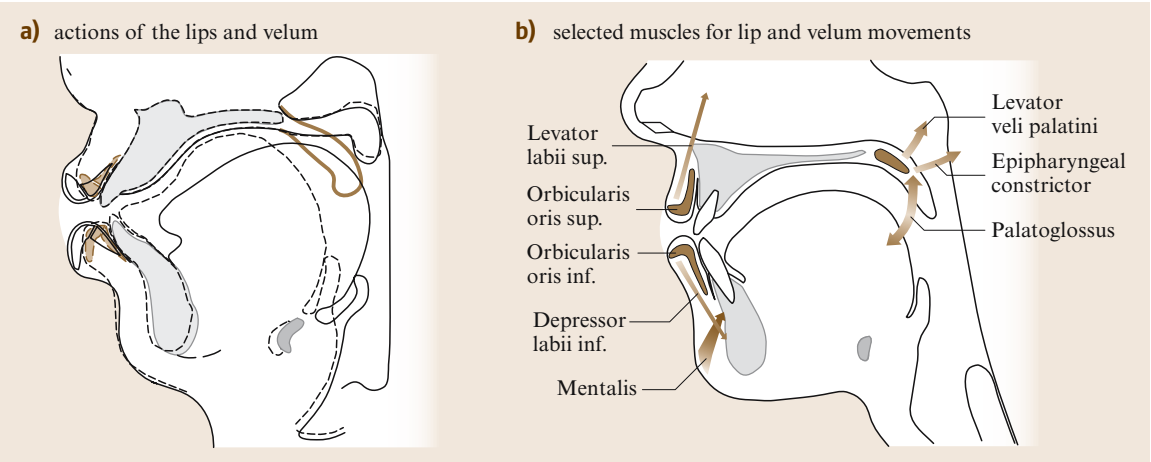


Fig. 2.12a,b Actions of the lips and velum, and their muscles. **(a)** Trace of MRI data in the production of /i/ and /u/ with lip protrusion show that two parts of the orbicularis oris, marginal (*front*) and peripheral (*back*) bundles demonstrate their geometrical changes within the vermillion tissue. The shapes of the velum also vary greatly between the rest position (thick gray line) and vowel articulation. **(b)** Five labial muscles are shown selectively from among many facial muscles. The velum shape is determined by the elevator, constrictor, and depressor (palatopharyngeus)

There are two muscles that support the tongue floor: the geniohyoid and mylohyoid muscles. The geniohyoid runs from the genial process of the mandibular symphysis to the body of the hyoid bone. This muscle has two functions: opening the jaw for open vowels and advancing the hyoid bone to help raise F_0. The mylohyoid is a sheet-like muscle beneath the tongue body that stretches between the mandible and the hyoid bone to support the entire tongue floor. This muscle supports the tongue floor to assist articulation of high front vowels and oral consonants.

Lips and Velum

The lips are a pair of soft-tissue organs consisting of many muscles. Their functions resemble those of the tongue because they partly adhere to the mandible and partly run within the soft tissue of the lips. The vermillion, or the part of red skin, is the unique feature of the human lips, which transmits phonetic signals visually. The deformation of the lips in speech can be divided into three components. The first is opening/closing of the lip aperture, which is augmented by jaw movement. The second is rounding/spreading of the lip tissue, produced by the changes in their left–right dimension. The third is protrusion/retraction of the lip gesture, generated by three-dimensional deformation of the entire lip tissue.

The muscles that cause deformation of the lips are numerous. Figure 2.12 shows only a few representative muscles of the lips. The orbicularis oris is the muscle that surrounds the lips, consisting of two portions; the marginal and peripheral bundles. Contraction of the marginal bundles near the vermillion borders is thought to produce lip rounding without protrusion. Contraction of the peripheral bundles that run in the region around the marginal bundles compresses the lip tissue circumferentially to advance the vermillion in lip protrusion [2.15]. The mentalis arises from the mental part of the mandible to the lip surface, and its contraction elevates the lower lip by pulling the skin at the mental region. The levator labii superior elevates the upper lip, and the depressor labii inferior depresses the lower lip relative to the jaw. The superior and inferior angli oris muscles move the lip corners up and down, respectively, which makes facial expressions rather than speech articulation.

The exact mechanism of lip protrusion is still in question. Tissue bunching by muscle shortening as a general rule for the organs of muscle does not fully apply to the phenomenon of lip protrusion. This is because, as the vermilion thickens in lip protrusion, it does not compress on the teeth; its dental surface often detaches from the teeth (Figure 2.12a). A certain three-dimensional stress distribution within the entire labial tissue must be considered to account for the causal factors of lip protrusion.

The velum, or the soft palate, works as a valve behind the hard palate to control the velopharyngeal port, as shown in Fig. 2.12a. Elevation of the velum is carried

out during the production of oral sounds, while lowering takes place during the production of nasal sounds. The action of the velum to close the velopharyngeal port is not a pure hinge motion but is accompanied by the deformation of the velum tissue with narrowing of the nasopharyngeal wall. In velopharyngeal closure, the levator veli palatine contracts to elevate the velum, and the superior pharyngeal constrictor muscle produces concentric narrowing of the port. In velopharyngeal opening, the palatoglossus muscle assists active lowering of the velum.

2.3.2 Vocal Tract and Nasal Cavity

The vocal tract is an acoustic space where source sounds for speech propagate. Vowels and consonants rely on strengthening or weakening of the spectral components of the source sound by resonance of the air column in the vocal tract. In the broad definition, the vocal tract includes all the air spaces where acoustic pressure variation takes place in speech production. In this sense, the vocal tract divides into three regions: the subglottal tract, the tract from the glottis to the lips, and the nasal cavities.

The subglottal tract is the lower respiratory tract below the glottis down to the lungs via the trachea and bronchial tubes. The length of the trachea from the glottis to the carina is 10–15 cm in adults, including the length of the subglottic laryngeal cavity (about 2 cm). Vocal source sounds propagate from the glottis to the trachea, causing the subglottal resonance in speech spectra. The resonance frequencies of the subglottal airway are estimated to be 640, 1400, and 2100 Hz [2.16]. The second subglottal resonance is often observed below the second formant of high vowels.

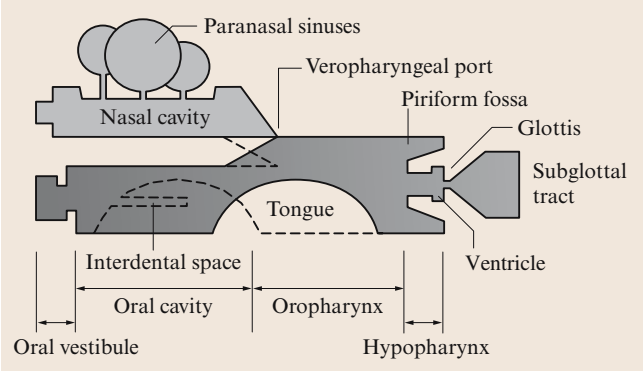


Fig. 2.13 Acoustic design of the vocal tract. Passages from the subglottal tract to two output ends at the lips and nares are shown with the effects of tongue and velar movements. The resonance of the vocal tract above the supraglottic laryngeal cavity determines major the vowel formants (F_1, F_2, and F_3). The resonance of the subglottal tract and interdental space interacts with the vowel formants, while the hypopharyngeal cavities and other small cavities cause local resonances and antiresonances in the higher-frequency region

The vocal tract, according to the conventional definition, is the passage of vocal sounds from the glottis to the lips, where source sounds propagate and give rise to the major resonances. The representative values for the length of the main vocal tract from the glottis to the lips are 15 cm in adult females and 17.5 cm in adult males. According to the measurement data based on the younger population, vocal tract lengths are 14 cm in females and 16.5 cm in males [2.17, 18], which are shorter than the above values. Considering the elongation of the vocal tract during a course of life, the above representative values appear reasonable. While the oral cavity length is maintained by the rigid structures of the skull and jaw, the pharyngeal cavity length increases due to larynx lowering before and after puberty. Thus, elongation of the pharyngeal cavity is the major factor in the developmental variation in vocal tract length.

The vocal tract anatomically divides into four segments: the hypopharyngeal cavities, the mesopharynx, the oral cavity, and the oral vestibule (lip tube). The hypopharyngeal part of the vocal tract consists of the supraglottic laryngeal cavity (2 cm long) and the bilateral conical cavities of the piriform fossa (2 cm long). The mesopharynx extends from the aryepiglottic fold to the anterior palatal arch. The oral cavity is the segment from the anterior palatal arch to the incisors. The oral vestibule extends from the incisors to the lip opening. The latter shows an anterior convexity, which often makes it difficult to measure the exact location of lip opening.

The vocal tract is not a simple uniaxial tube but has a complex three-dimensional construction. The immobile wall of the vocal tract includes the dental arch and the palatal dome. The posterior pharyngeal wall is almost rigid, but it allows subtle changes in convexity and orientation. The soft walls include the entire tongue surface, the velum with the uvula, the lateral pharyngeal wall, and the lip tube. The shape of the vocal tract varies individually due to a few factors. First, the lateral width of the upper and lower jaws relative to the pharyngeal cavity width affects tongue articulation and results in a large individual variation of vocal tract shape observed midsagittally. Second, the mobility of the jaw

depending on the location of the mandibular symphysis relative to the skull can vary the openness of vowels. Third, the size of the tongue relative to the oral and pharyngeal cavities varies individually; the larger the tongue size, the smaller the articulatory space for vowels.

Figure 2.13 shows a schematic drawing of the vocal tract and nasal cavity. The vocal tract has nearly constant branches such as the piriform fossa (entrance to the esophagus) and the vallecula (between the tongue root and epiglottis). The vocal tract also has controlled branches to the nasal cavity at the velopharyngeal port and to the *interdental space* (the space bounded by the upper and lower teeth and the lateral cheek wall). The latter forms a pair of side-branches when the tongue is in a higher position as in /i/ or /e/, while it is unified with the oral cavity when the tongue is in a lower position as in /a/.

The nasal cavity is an accessory channel to the main vocal tract. Its horizontal dimension from the anterior nares to the posterior wall of the epipharynx is approximately 10–11 cm. The nasal cavity can be divided into the single-tube segment (the velopharyngeal region and epipharynx) and the dual-tube segment (the nasal cavity proper and nasal vestibule). Each of the bilateral channels of the nasal cavity proper has a complex shape of walls with the three turbinates with thick mucous membrane, which makes a narrower cross section compared with the epipharyngeal area [2.19]. The nasal cavity has its own side-branches of the paranasal sinuses; the maxillary, sphenoid, ethomoid, and frontal sinuses.

The nasal cavity builds nasal resonance to accomplish phonetic features of nasal sounds and nasalized vowels. The paranasal sinuses also contribute to acoustic characteristics of the nasal sounds. The nasal murmur results from these characteristics: a Helmholtz resonance of the entire nasopharyngeal tract from the glottis to the anterior nares and regional Helmholtz resonances caused by the paranasal sinuses, together characterized by a resonance peak at 200–300 Hz and spectral flattening up to 2 kHz [2.20, 21]. The nasal resonance could takes place even in oral vowels with a complete closure of the velopharyngeal port: the soft tissue of the velum transmits the pressure variation in the oral cavity to the nasal cavity, which would enhances sound radiation for close vowels and voiced stops.

2.3.3 Aspects of Articulation in Relation to Voicing

Here we consider a few phonetic evidences that can be considered as joint products of articulation and phonation. Vowel production is the typical example for this topic, in view of its interaction with the larynx. Regulation of voice quality, which has been thought to be a laryngeal phenomenon, is largely affected by the lower part of the vocal tract. The voiced versus voiceless distinction is a pertinent issue of phonetics that involves both phonatory and articulatory mechanisms.

Production of Vowels

The production of vowels is the result of the joint action of phonatory and articulatory mechanisms. In this process, the larynx functions as a source generator, and the vocal tract plays the role of an acoustic filter to modulate the source sounds and radiate from the lip opening, as described by the *source-filter* theory [2.22, 23]. The quality of oral vowels is determined by a few peak frequencies of vocal tract resonance (formants). In vowel production, the vocal tract forms a *closed tube* with the closed end at the glottis and the open end at the lip opening. Multiple reflections of sound wave between the two ends of the vocal tract give rise to vowel formants (F_1, F_2, F_3). The source-filter theory has been supported by many studies as the fundamental concept explaining the acoustic process of speech production, which is further discussed in the next section.

Vowel articulation is the setup for the articulatory organs to determine vocal tract shape for each vowel. When the jaw is in a high position and the tongue is in a high front position, the vocal tract assumes the shape for /i/. Contrarily, when the jaw is in a low position and the tongue is in a low back position, the vocal tract takes the shape for /a/. The articulatory organ that greatly influences vocal tract shape for vowels is the tongue. When the vocal tract is modeled as a tube with two segments (front and back cavities), the movements of the tongue body between its low back and high front positions creates contrasting diverging and converging shapes of the main vocal tract. Jaw movement enhances these changes in the front cavity volume, while pharyngeal constriction assists in the back cavity volume. When the vocal tract is modeled as a tube with three segments, the movements of the tongue body between its high back and low front positions determine the constriction or widening of the vocal tract in its middle portion. The velum also contributes to the articulation of open vowels by decreasing the area of the vocal tract at the velum or making a narrow branch to the nasal cavity. The lip tube is another factor for vowel articulation that determines the vocal tract area near the open end.

Although muscular control for vowel articulation is complex, a simplified view can be drawn based

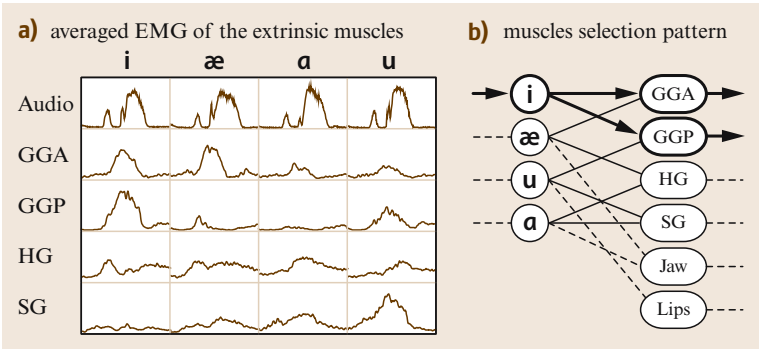


Fig. 2.14a,b Tongue EMG data during VCV utterances and muscle selection pattern in vowel articulation. **(a)** Averaged EMG data for four English corner vowels are shown for the major muscles of the tongue: the anterior genioglossus (GGA), posterior genioglossus (GGP), hyoglossus (HG), and styloglossus (SG). **(b)** The systematic variation observed in the muscle–vowel matrix suggests a muscle selection pattern

on electromyographic (EMG) data obtained from the tongue muscles [2.24]. Figure 2.14a shows a systematic pattern of muscle activities for CVC (consonant-vowel-consonant) utterances with /p/ and four English corner vowels. The anterior and posterior genioglossus are active for front vowels, while the styloglossus and hyoglossus are active for back vowels. These muscles also show a variation depending on vowel height. These observations are shown schematically in Fig. 2.14b: the basic control pattern for vowel articulation is the selection of two muscles among the four extrinsic muscles of the tongue [2.25].

As the tongue or jaw moves for vowel articulation, they apply forces to the surrounding organs and cause secondary effects on vowel sounds. For example, articulation of high vowels such as /i/ and /u/ is mainly produced by contraction of the posterior genioglossus, which is accompanied by forward movement of the hyoid bone. This action applies a force to rotate the thyroid cartilage in a direction that stretches the vocal folds. In evidence, higher vowels tend to have a higher F_0, known as the *intrinsic vowel* F_0 [2.26,27]. When the jaw opens to produce open vowels, jaw rotation compresses the tissue behind the mandibular symphysis, which applies a force to rotate the thyroid cartilage in the opposite direction, thereby shortening the vocal folds. Thus, the jaw opening has the secondary effect of lowering the intrinsic F_0 for lower vowels.

Supra-Laryngeal Control of Voice Quality

The laryngeal mechanisms controlling voice quality were described in an earlier section. In this section, the supra-laryngeal factors are discussed. Recent studies have shown evidence that the resonances of the hypopharyngeal cavities determine the spectral envelope in the higher frequencies above 2.5 kHz by causing an extra resonance and antiresonances [2.28–31]. The hypopharyngeal cavities include a pair of vocal-tract

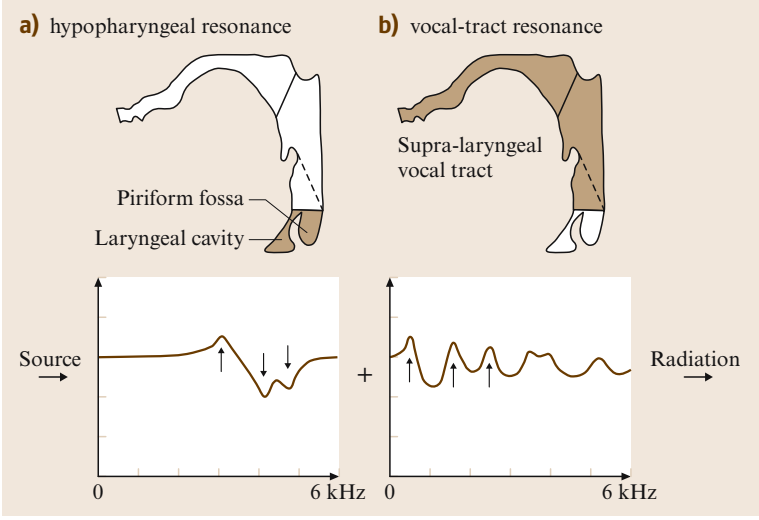


Fig. 2.15a,b Vocal-tract resonance with hypopharyngeal cavity coupling in vowel production. **(a)** The supraglottal laryngeal cavity contributes a resonance peak at 3–3.5 kHz, and the bilateral cavities of the piriform fossa cause antiresonances at 4–5 kHz. **(b)** The main vocal tract above the laryngeal cavity determines the major vowel formants

side-branches formed by the piriform fossa. Each fossa maintains a relatively constant cavity during speech, which is collapsed only in deep inhalation by the wide abduction of the arytenoid cartilage. The piriform fossa causes one or two obvious antiresonances in the higher frequencies above 4 kHz [2.29] and affects the surrounding formants. The laryngeal cavity above the vocal folds also contributes to shaping the higher frequencies [2.28, 32]. The supraglottic laryngeal cavity, from the ventricles to the aryepiglottic folds via the ventricular folds, forms a type of Helmholtz resonator and gives rise to a resonance at higher frequencies of 3–3.5 kHz. This resonance can be counted as the fourth formant (F4) but it is actually an *extra formant* to the resonance of the vocal tract above the laryngeal cavity [2.30]. When the glottis opens in the open phase of vocal fold vibration, the supraglottic laryngeal cavity no longer constitutes a typical Helmholtz resonator, and demonstrates a strong damping of the resonance, which is observed as the disappearance of the affiliated extra formant. Therefore, the laryngeal cavity resonance shows a cyclic nature during vocal fold vibration, and it is possibly absent in breathy phonation or pathological conditions with insufficient glottal closure [2.31]. Figure 2.15 shows an acoustic model of the vocal tract to illustrate this coupling of the hypopharyngeal cavities.

The hypopharyngeal cavities are not an entirely fixed structure but vary due to physiological efforts to control F_0 and voice quality. A typical case of the hypopharyngeal adjustment of voice quality is found in the *singing formant* [2.28]. When high notes are produced by opera singers, the entire larynx is pulled forward due to the advanced position of the tongue, which widens the piriform fossa to deepens the fossa's antiresonances, resulting in a decrease of the frequency of the adjacent lower formant (F_5). When the supraglottic laryngeal cavity is constricted, its resonance (F_4) comes down towards the lower formant (F_3). Consequently, the third to fifth formants come closer to each other and generate a high resonance peak observed near 3 kHz.

Regulation of Voiced and Voiceless Sounds

Voiced and voiceless sounds are often attributed to the glottal state with and without vocal fold vibration, while their phonetic characteristics result from phonatory and articulatory controls over the speech production system. In voiced vowels, the vocal tract forms a closed tube with no significant constrictions except for the narrow laryngeal cavity. On the other hand, in whispered vowels, the membranous glottis is closed, and the supraglottic laryngeal cavity forms an extremely narrow channel continued from the open cartilaginous glottis, with a moderate constriction of the lower pharynx. Devoiced vowels exhibit a wide open glottis and a reduction of tongue articulation. Phonetic distinctions of voiced and voiceless consonants further involve fine temporal control over the larynx

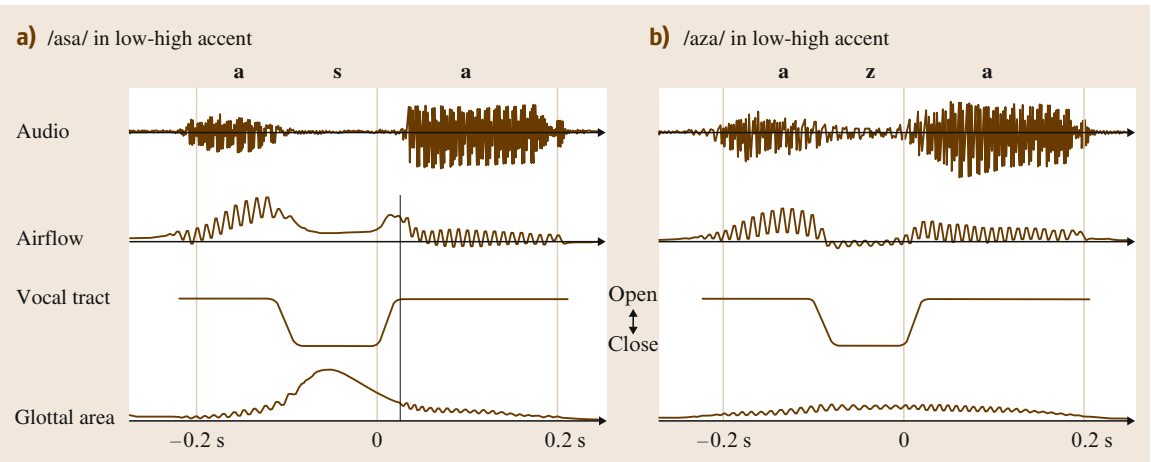


Fig. 2.16a,b Laryngeal articulatory patterns in producing VCV utterances with voiceless and voiced fricatives as in /asa/ and /aza/. From the top to bottom, speech signals, oral airflow, schematic patterns of vocal tract constriction, and glottal area variations are shown schematically. This figure is based on the author's recent experiment with anemometry with an open-type airflow transducer and photoglotto-graphy with an external lighting technique, conducted by *Dr. Shinji Maeda* (ENST) and the author

and supra-laryngeal articulators in language-specific ways.

In the production of voiced consonants, vocal fold vibration typically continues during the voiced segments. In voiced stops and fricatives, the closure or narrowing of the vocal tract results in decrease in glottal airflow and transglottal pressure difference. The glottal airflow during the stop closure is maintained during the closure due to the increases in vocal tract volume: the expansion of the oral cavity (jaw lowering and cheek wall expansion) and the expansion of the pharyngeal cavity (lateral wall expansion and larynx lowering). During the closure period, air pressure variations are radiated not only from the vocal tract wall but also from the anterior nares due to transvelar propagation of the intra-oral sound pressure into the nasal cavities.

In the production of voiceless consonants, vocal fold vibration is suppressed due to a rapid reduction of the transglottal pressure difference and abduction of the vocal folds. During stop closures, the intra-oral pressure builds up to reach the subglottal pressure, which enhances the rapid airflow after the release of the closure. Then, vocal fold vibration restarts with a delay to the release, which is observed as a long voice onset time (VOT) for voiceless stops. The process of suppressing vocal fold vibration is not merely a passive aerodynamic process on the vocal folds, but is assisted by a physiological process to control vocal fold stiffness. The cricothyroid muscle has been observed to increase its activity in producing voiceless consonants. This activity results in a high–falling F_0 pattern during the following vowel, contributing a phonetic attribute to voiceless consonants [2.33]. In glottal stops, vocal fold vibration stops due to forced adduction of the vocal folds with an effort closure of the supraglottic laryngeal cavity.

Figure 2.16 illustrates the time course of the processes during vowel-consonant-vowel (VCV) utterances with a voiceless fricative in comparison to the case with a voiced fricative. The voiceless segment initiates with glottal abduction and alveolar constriction, and vocal fold vibration gradually fades out during the phase of glottal opening. After reaching the maximum glottal abduction, the glottis enters the adduction phase, followed by the release of the alveolar constriction. Then, the glottis becomes narrower and vocal fold vibration restarts. There is the time lag between the release of the constriction and full adduction of the glottis, which results in the peak flow seen in Fig. 2.16a, presumably accompanied by aspiration sound at the glottis.

2.3.4 Articulators' Mobility and Coarticulation

The mobility of speech articulators varies across organs and contributes certain phonetic characteristics to speech sounds. Rapid movements are essential to a sequence from one distinctive feature to another, as observed in the syllable /sa/ from a narrow constriction to the vocalic opening, while gradual movements are found to produce nasals and certain labial sounds. These variations are principally due to the nature of articulators with respect to their mobility. The articulatory mechanism involves a complex system that is built up by organs with different motor characteristics. Their variation in temporal mobility may be explained by a few biological factors. The first is the phylogenetic origin of the organs: the tongue muscles share their origin with the fast motor systems such as the eyeball or finger, while other muscles such as in the lips or velum originate from the slow motor system similar to the musculature of the alimentary tract. The second is the innervation density to each muscle: the faster organs are innervated by thicker nerve bundles, and vise versa, which derives from an adaptation of the biological system to required functions. In fact, the human hypoglossal nerve that supplies the tongue muscles is much thicker than that of other members of the primate family. The third is the composition of muscle fiber types in the musculature, which varies from organ to organ. The muscles in the larynx have a high concentration of the ultrafast fibers (type 2B), while the muscle to elevate the velum predominantly contains the slow fibers (type 1). In accordance with these biological views, the rate of the articulators movement indexed by the maximum number of syllables per second follows the order of the tongue apex, body, and lips: the tongue moves at a maximum rate of 8.2 syllables per second at the apex, and 7.1 syllables per second with the back of the tongue, while the lips and facial structures move at a maximum rate of 2.5–3 syllables per second [2.34]. More recent measurements indicate that the lips are slower than the tongue apex but faster than the tongue dorsum. The velocities during speech tasks reach 166 mm/sec for the lower lip, 196 mm/sec for the tongue tip, and 129 mm/sec for the tongue dorsum [2.35]. The discrepancy between these two reports regarding the mobility of the lips may be explained by the types of movements measured: opening–closure movement by the jaw–lower lip complex is faster than the movement of the lips themselves, such as protrusion and spreading.

It is often noted that speech is characterized by asynchrony among articulatory movements, and the degree

of asynchrony varies with the feature to be realized. Each articulator does not necessarily strictly keep pace with other articulators in a syllable sequence. The physiological basis of this asynchrony may be explained by the mobility of the articulatory organs and motor precision required for the target of articulation. The slower articulators such as the lips and velum tend to exhibit marked coarticulation in production of labial and nasal sounds. In stop–vowel–nasal sequences (such as /tan/), the velopharyngeal port is tightly closed at the stop onset and the velum begins to lower before the nasal consonant. Thus, the vowel before the nasal consonant is partly nasalized. When the vowel /u/ is preceded by /s/, the lips start to protrude during the consonant prior to the rounded vowel.

The articulators' mobility also contributes some variability to speech movements. The faster articulators such as parts of the tongue show various patterns from target undershooting to overshooting. In articulation of close–open–close vowel sequences such as /iai/, tongue movements naturally show undershooting for the open vowel. In contrast, when the alveolar voiceless stop /t/ is placed in the open vowel context as in /ata/, the tongue blade sometimes shows an extreme overshoot with a wide contact on the hard palate because such articulatory variations do not significantly affect the output sounds. On the contrary, in alveolar and postalveolar fricatives such as /s/ and /sh/, tongue movements also show a dependence on articulatory precision because the position of the tongue blade must be controlled precisely to realize the narrow passage for generating frication sounds. The lateral /l/ is similar to the stops with respect to the palatal contact, while the rhotic /r/ with no contact to the palate can show a greater extent of articulatory variations from retroflex to bunched types depending on the preceding sounds.

2.3.5 Instruments for Observing Articulatory Dynamics

X-ray and palatography have been used as common tools for articulatory observation. Custom instruments are also developed to monitor articulatory movements, such as the X-ray microbeam system and magnetic sensor system. The various types of newer medical imaging techniques are being used to visualize the movements of articulatory system using sonography and nuclear magnetic resonance. These instruments are generally large scale, although relatively compact instruments are becoming available (e.g., magnetic probing system or portable ultrasound scanner).

Palatography

The palatograph is a compact device to record temporal changes in the contact pattern of the tongue on the palate. There are traditional static and modern dynamic types. The dynamic type is called electropalatography, or dynamic palatography, which employs an individually customized palatal plate to be placed on the upper jaw. As shown in Fig. 2.17a, this system employs a palatal plate with many surface electrodes to monitor electrical contacts on the tongue's surface.

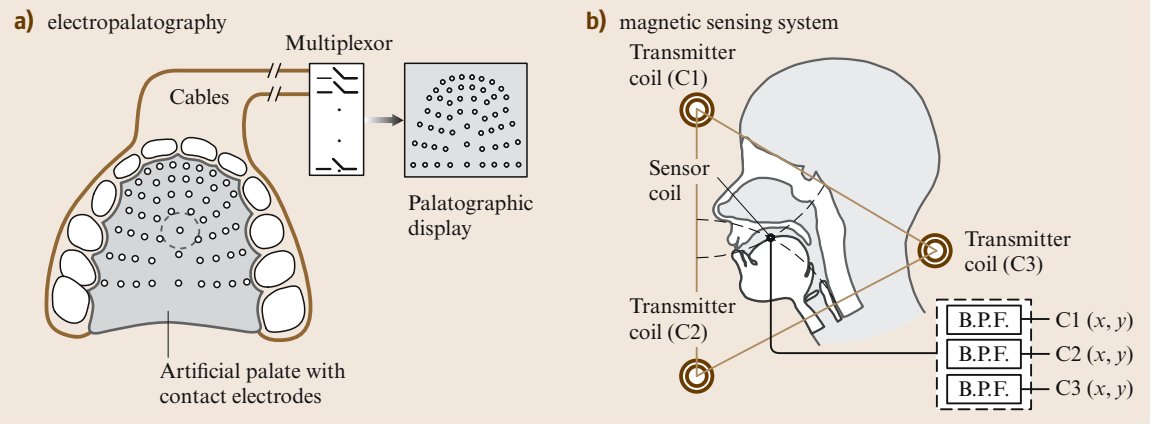


Fig. 2.17a,b Electropalatography and magnetic sensing system. **(a)** Electropalatography displays tongue–palate contact patterns by detecting weak electrical current caused by the contact between the electrodes on the artificial palate and the tongue tissue. **(b)** Magnetic sensing system is based on detection of alternate magnetic fields with different frequencies using miniature sensor coils

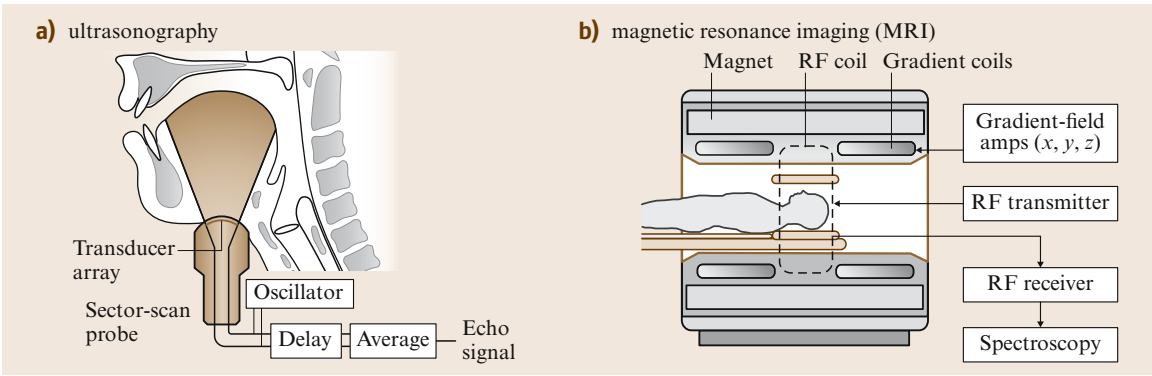


Fig. 2.18a,b Medical imaging techniques. **(a)** Ultrasound scanner uses an array of transmitters and receivers to detect echo signals from regions where the ultrasound signals reflect strongly such as at the tissue-air boundaries on the tongue surface. **(b)** Magnetic resonance imaging (MRI) generates strong static magnetic field, controlled gradient fields in the three directions, and radio-frequency (RF) pulses. Hydrogen atoms respond to the RF pulses to generate echo signals, which are detected with a receiver coil for spectral analysis

Marker Tracking System

A few custom devices have been developed to record movements of markers attached on the articulatory organs. X-ray microbeam and magnetic sensing systems belong to this category. Both can measure 10 markers simultaneously. The X-ray microbeam system uses a computer-controlled narrow beam of high-energy X-rays to track small metal pellets attached on the articulatory organs. This system allows automatic accurate measurements of pellets with a minimum X-ray dosage.

The magnetic sensing system (magnetometer, or magnetic articulograph) is designed to perform the same function as the microbeam system without X-rays. The system uses a set of transmitter coils that generate alternate magnetic fields and miniature sensor coils attached to the articulatory organs, as shown in Fig. 2.17b. The positions of the receiver coils are computed from the filtered signals from the coils.

Medical Imaging Techniques

X-ray cinematography and X-ray video fluorography have been used for re-cording articulatory movements in two-dimensional projection images. The X-ray images show clear outlines of rigid structures, while they provide less-obvious boundaries for soft tissue. The outline of the tongue is enhanced by the application of liquid contrast media on the surface. Metal markers are often used to track the movements of flesh points on the soft-tissue articulators.

Ultrasonography is a diagnostic technique to obtain cross-sectional images of soft-tissues in real time. Ultrasound scanners consist of a sound probe (phased-array piezo transducer and receiver) and image processor, as illustrated in Fig. 2.18a. The probe is attached to the skin below the tongue to image the tongue surface in the sagittal or coronal plane.

Magnetic resonance imaging (MRI), shown in Fig. 2.18b, is a developing medical technique that excels at soft-tissue imaging of the living body. Its principle relies on excitation and relaxation of the hydrogen nuclei in water in a strong homogeneous magnetic field in response to radio-frequency (RF) pulses applied with variable gradient magnetic fields that determine the slice position. MRI is essentially a method for recording static images, while motion imaging setups with stroboscopic or real-time techniques have been applied to the visualization of articulatory movements or vocal tract deformation three-dimensionally [2.36].

2.4 Summary

This chapter described the structures of the human speech organs and physiological mechanisms for producing speech sounds. Physiological processes during speech are multidimensional in nature as described in this chapter. Discoveries of their component mechanisms have been dependent on technical developments

for visualizing the human body and analyses of biological signals, and this is still true today. For example, the hypopharyngeal cavities have long been known to exist, but their acoustic role was underestimated until recent MRI observations. The topics in this chapter were chosen with the author's hope to provide a guideline for the sophistication of speech technologies by reflecting the real and detailed processes of human speech production. Expectations from these lines of studies include speech analysis by recovering control parameters of articulatory models from speech sounds, speech synthesis with full handling of voice quality and individual vocal characteristics, and true speech recognition through biologic, acoustic, and phonetic characterizations of input sounds.

References

2.1 M.H. Draper, P. Ladefoged, D. Whittenridge: Respiratory muscles in speech, J. Speech Hearing Res. **2**, 16–27 (1959)

2.2 T.J. Hixon, M. Goldman, J. Mead: Kinematics of the chest wall during speech production: volume displacements of the rib cage, abdomen, and lung, J. Speech Hearing Res. **16**, 78–115 (1973)

2.3 G. Weismer: Speech production. In: *Handbook of Speech-Language Pathology and Audiology*, ed. by N.J. Lass, L.V. McReynolds, D.E. Yoder (Decker, Toronto 1988) pp. 215–252

2.4 J. Kahane: A morphological study of the human prepubertal and pubertal larynx, Am. J. Anat. **151**, 11–20 (1979)

2.5 M. Hirano, Y. Kakita: Cover-body theory of vocal cord vibration. In: *Speech Science*, ed. by R. Daniloff (College Hill, San Diego 1985) pp. 1–46

2.6 E.B. Holmberg: Glottal airflow and transglottal air pressure measurements for male and female speakers in soft, normal, and loud voice, J. Acoust. Soc. Am. **84**, 511–529 (1988)

2.7 M.R. Rothenberg: Acoustic interaction between the glottal source and the vocal tract. In: *Vocal Fold Physiology*, ed. by K.N. Stevens, M. Hirano (Univ. Tokyo Press, Tokyo 1981) pp. 305–328

2.8 G. Fant, J. Liljencrants, Q. Lin: A four-parameter model of glottal flow, Speech Transmission Laboratory – Quarterly Progress and Status Report (STL-QPSR) **4**, 1–13 (1985)

2.9 B.R. Fink, R.J. Demarest: *Laryngeal Biomechanics* (Harvard Univ. Press, Cambridge 1978)

2.10 J.E. Atkinson: Correlation analysis of the physiological features controlling fundamental frequency, J. Acoust. Soc. Am. **63**, 211–222 (1978)

2.11 K. Honda, H. Hirai, S. Masaki, Y. Shimada: Role of vertical larynx movement and cervical lordosis in F0 control, Language Speech **42**, 401–411 (1999)

2.12 H. Takemoto: Morphological analysis of the human tongue musculature for three-dimensional modeling, J. Speech Lang. Hearing Res. **44**, 95–107 (2001)

2.13 S. Takano, K. Honda: An MRI analysis of the extrinsic tongue muscles during vowel production, Speech Commun. **49**, 49–58 (2007)

2.14 S. Maeda: Compensatory articulation during speech: evidence from the analysis and synthesis of vocal-tract shapes using an articulatory model. In: *Speech Production and Speech Modeling*, ed. by W.J. Hardcastle, A. Marchal (Kluwer Academic, Dartrecht 1990) pp. 131–149

2.15 K. Honda, T. Kurita, Y. Kakita, S. Maeda: Physiology of the lips and modeling of lip gestures, J. Phonetics **23**, 243–254 (1995)

2.16 K. Ishizaka, M. Matsudaira, T. Kaneko: Input acoustic-impedance measurement of the subglottal system, J. Acoust. Soc. Am. **60**, 190–197 (1976)

2.17 U.G. Goldstein: An articulatory model for the vocal tracts of growing children. Ph.D. Thesis (Massachusetts Institute of Technology, Cambridge 1980)

2.18 W.T. Fitch, J. Giedd: Morphology and development of the human vocal tract: A study using magnetic resonance imaging, J. Acoust. Soc. Am. **106**, 1511–1522 (1999)

2.19 J. Dang, K. Honda, H. Suzuki: Morphological and acoustic analysis of the nasal and paranasal cavities, J. Acoust. Soc. Am. **96**, 2088–2100 (1994)

2.20 O. Fujimura, J. Lindqvist: Sweep-tone measurements of the vocal tract characteristics, J. Acoust. Soc. Am. **49**, 541–557 (1971)

2.21 S. Maeda: The role of the sinus cavities in the production of nasal vowels, Proc. IEEE Int. Conf. Acoustics, Speech, Signal Proc. (ICASSP'82), Vol. 2 (1982) pp. 911–914, Paris

2.22 T. Chiba, M. Kajiyama: *The Vowel – Its Nature and Structure* (Tokyo-Kaiseikan, Tokyo 1942)

2.23 G. Fant: *Acoustic Theory of Speech Production* (Mouton, The Hague 1960)

2.24 T. Baer, P. Alfonso, K. Honda: Electromyography of the tongue muscle during vowels in /*pVp/ environment, Ann. Bull. RILP **22**, 7–20 (1988)

2.25 K. Honda: Organization of tongue articulation for vowels, J. Phonetics **24**, 39–52 (1996)

2.26 I. Lehiste, G.E. Peterson: Some basic considerations in the analysis of intonation, J. Acoust. Soc. Am. **33**, 419–425 (1961)

2.27 K. Honda: Relationship between pitch control and vowel articulation. In: *Vocal Fold Physiology*, ed. by D.M. Bless, J.H. Abbs (College-Hill, San Diego 1983) pp. 286–297

2.28 J. Sundberg: Articulatory interpretation of the singing formant, J. Acoust. Soc. Am. **55**, 838–844 (1974)

2.29 J. Dang, K. Honda: Acoustic characteristics of the piriform fossa in models and humans, J. Acoust. Soc. Am. **101**, 456–465 (1996)
2.30 H. Takemoto, S. Adachi, T. Kitamura, P. Mokhtari, K. Honda: Acoustic roles of the laryngeal cavity in vocal tract resonance, J. Acoust. Soc. Am. **120**, 2228–2238 (2006)
2.31 T. Kitamura, H. Takemoto, S. Adachi, P. Mokhtari, K. Honda: Cyclicity of laryngeal cavity resonance due to vocal fold vibration, J. Acoust. Soc. Am. **120**, 2239–2249 (2006)
2.32 I.R. Titze, B.H. Story: Acoustic interactions of the voice source with the lower vocal tract, J. Acoust. Soc. Am. **101**, 2234–2243 (1997)
2.33 A. Lofqvist, N.S. McGarr, K. Honda: Laryngeal muscles and articulatory control, J. Acoust. Soc. Am. **76**, 951–954 (1984)
2.34 R.G. Daniloff: Normal articulation processes. In: *Normal Aspect of Speech, Hearing, and Language*, ed. by F.D. Minifie, T.J. Hixon, F. Williams (Prentice-Hall, Englewood Cliffs 1983) pp. 169–209
2.35 D.P. Kuehn, K.L. Moll: A cineradiographic study of VC and CV articulatory velocities, J. Phonetics **4**, 303–320 (1976)
2.36 K. Honda, H. Takemoto, T. Kitamura, S. Fujita, S. Takano: Exploring human speech production mechanisms by MRI, IEICE Info. Syst. **E87-D**, 1050–1058 (2004)

5. Speech Quality Assessment

V. Grancharov, W. B. Kleijn

In this chapter, we provide an overview of methods for speech quality assessment. First, we define the term *speech quality* and outline in Sect. 5.1 the main causes of degradation of speech quality. Then, we discuss subjective test methods for quality assessment, with a focus on standardized methods. Section 5.3 is dedicated to objective algorithms for quality assessment. We conclude the chapter with a reference table containing common quality assessment scenarios and the corresponding most suitable methods for quality assessment.

The rapid deployment of speech processing applications increases the need for speech quality evaluation. The success of any new technology (e.g., network equipment, speech codec, speech synthesis system, etc.) depends largely on end-user opinion of perceived speech quality. Therefore, it is vital for the developers of a new service or speech processing application to assess its speech quality on a regular basis.

In addition to its role for services and speech processing, speech quality evaluation is of critical importance in the areas of clinical hearing diagnostics and psychoacoustical research. Although this chapter addresses speech quality mainly from the viewpoint of telecommunication applications, it is also of general interest for researchers dealing with speech quality assessment methods.

When the speech signal reaches the human auditory system, a speech perception process is initiated. This process results in an *auditory event*, which is internal and can be measured only through a description by the listener (the subject). The subject then establishes a relationship between the perceived and expected auditory event. Thus, the speech quality is a result of a perception and assessment process.

Since the quality of a speech signal does not exist independently of a subject, it is a *subjective* measure. The most straightforward manner to estimate speech quality is to play a speech sample to a group of listeners, who are asked to rate its quality. Since subjective quality assessment is costly and time consuming, computer algorithms are often used to determine an *objective* quality measure that approximates the subjective rating. Section 5.2 provides an overview of subjective tests for speech quality assessment, while Sect. 5.3 is dedicated to objective quality assessment measures.

Speech quality has many perceptual dimensions. Commonly used dimensions are intelligibility, naturalness, loudness, listening effort, etc., while less commonly used dimensions include nasality, graveness, etc. However, the use of a *multidimensional metric* for quality assessment is less common than the use of a *single metric*, mainly as a result of cost and complexity. A single metric, such as the mean opinion score scale, gives an integral (overall) perception of an auditory event and is therefore sufficient to predict the end-user opinion of a speech communication system. However, a single metric does not in general provide sufficient detail for system designers. Multidimensional-metric tests

are discussed in Sect. 5.2.2 and single-metric tests are discussed throughout the remainder of Sect. 5.2.

In some applications, it is desirable or historically accepted to measure only specific quality dimensions, such as *intelligibility*, *listening effort*, *naturalness*, and *ability for talker recognition*. The most popular among these measures are covered in Sect. 5.2.3.

The *true* speech quality is often referred to as *conversational quality*. Conversational tests usually involve communication between two people, who are questioned later about the quality aspects of the conversation, see Sect. 5.2.1. However, the most frequently measured quantity is *listening quality*, which is the focus of Sect. 5.2.1. In the listening context, the speech quality is mainly affected by speech distortion due to speech codecs, background noise, and packet loss. One can also distinguish *talking quality*, which is mainly affected by echo associated with delay and sidetone distortion.

The distorted (processed) signal or its parametric representation is always required in an assessment of speech quality. However, based on the availability of the original (unprocessed) signal, two test situations are possible: *reference based* and *not reference based*. This classification is common for both the subjective and objective evaluation of speech quality. The absolute category rating (ACR) procedure, popular in subjective tests, does not require the original signal, while in the degradation category rating (DCR) approach the original signal is needed. In objective speech quality assessment, the historically accepted terms are *intrusive* (with original) and *non-intrusive* (without original). These two test scenarios will be discussed throughout the chapter.

5.1 Degradation Factors Affecting Speech Quality

The main underlying causes of degradation of speech quality in modern speech communication systems are delay (latency), packet loss, packet delay variation (jitter), echo, and distortion introduced by the codec. These factors affect psychological parameters such as intelligibility, naturalness, and loudness, which in turn determine the overall speech quality.

In this section, we briefly list the most common impairment factors. We divide them into three classes:

1. factors that lead to listening difficulty
2. factors that lead to talking difficulty
3. factors that lead to conversational difficulty

The reader can find more-detailed information in International Telecommunication Union, Telecommunication Standardization Sector (ITU-T) Rec. G.113 [5.1]. The effect of transmission impairments on users is discussed in ITU-T Rec. P.11 [5.2].

Degradation factors that cause an increase in *listening difficulty* include packet loss, distortion due to speech codecs, speech clipping, and listener echo. *Packet loss* corresponds to the percentage of speech frames that do not reach their final destination. If no protective measures are taken, a packet loss rate of 5% results in significant degradation of the speech quality. Bursts of packet loss also affect speech quality. In systems without error concealment, *speech clipping* occurs at any time when the transmitted signal is lost. Speech clipping may temporarily occur when the connection suffers from packet loss or when voice activity detectors are used. *Listener echo* refers to a transmission condition in which the main speech signal arrives at the listener's end of the connection accompanied by one or more delayed versions (echoes) of the signal. The intelligibility decreases as the *loudness loss* increases. On the other hand, if the loudness loss decreases too much, customer satisfaction decreases because the received speech is too loud.

Degradation factors that cause difficulty while *talking* are talker echo and an incorrectly set sidetone. *Talker echo* occurs when some portion of the talker's speech signal is returned with a delay sufficient (typically more than 30 ms) to make the signal distinguishable from the normal sidetone. The *sidetone* of a telephone set is the transmission of sound from the telephone microphone to the telephone receiver in the same telephone set. Too little sidetone loss causes the returned speech levels to be too loud and thus reduces customer satisfaction. Excessive sidetone loss can make a telephone set sound *dead* as one is talking. In addition, the sidetone path provides another route by which room noise can reach the ear.

Conversation difficulties are caused by a third class of degradation factors. *Delay* is defined as the time it takes for the packet to arrive at its destination. Long delays impair a conversation. *Intelligible crosstalk* occurs when the speech signal from one telephone connection is coupled to another telephone connection such that the coupled signal is audible and intelligible to one or both of the participants on the second telephone connection. The *background noise* in the environment of the tele-

phone set may have a substantial effect on the ease of carrying on a conversation.

The study of degradation factors is important in the design of a speech quality assessment test. The set of degradation factors present in a communication system determines the type of test to be performed. If the degradation factors cause only an increase in listening difficulty, it is sufficient to perform relatively inexpensive and simple tests that measure *listening quality*. If the degradation factors cause difficulty while talking, or difficulty while conversing, it is recommended to perform the more-complex *conversational quality* tests.

5.2 Subjective Tests

Speech quality is a complex psychoacoustic outcome of the human perception process. As such, it is necessarily subjective, and can be assessed through listening test involving human test subjects that listen to a speech sample and assign a rating to it. In this section, we cover the most commonly used subjective quality tests.

5.2.1 Single Metric (Integral Speech Quality)

Users of new speech processing applications are often unaware of the underlying technology. Their main criterion for assessing these applications is based on overall speech quality. Therefore, we start our discussion with *single-metric* subjective tests. In these tests, speech is played to a group of listeners, who are asked to rate the quality of this speech signal based on their *overall perception*.

Listening Quality

In an ACR test, a pool of listeners rate a series of audio files using a five-level impairment scale, as shown in Table 5.1. After each sample is heard, the listeners express an opinion, based only on the most recently heard sample. The average of all scores thus obtained for speech produced by a particular system represents its *mean opinion score* (MOS). The ACR listening quality method is standardized in [5.3], and is the most commonly used subjective test procedure in telecommunications. The main reason for the popularity of this test is its simplicity.

A good method for obtaining information on the detectability of a distortion (e.g., echo) as a function of some objective quantity (e.g., listening level) is to use the *detectability opinion scale* (Table 5.2). The decisions on a detectability scale are not equivalent to responses on a continuous scale. It is therefore recommended to use as a method of analysis the probability of response [5.3].

A disadvantage of ACR methods is that for some applications the resolution of their quality scale is not sufficient. In such cases the DCR method is appropriate. DCR methods provide a quality scale of higher resolution, due to comparison of the distorted signal with one or more reference/anchor signals. In a DCR test, the listeners are presented with the unprocessed signal as a reference before they listen to the processed signal. The task for the listener is to rate the perceived degradation by comparing the second stimulus to the first on the scale presented in Table 5.3. The quantity evaluated from the scores is referred to as the *degradation mean opinion score* (DMOS). DCR methods are also standardized in [5.3].

ABX is another popular method for speech quality assessment [5.4]. It consists of presenting the listener

Table 5.1 Grades in the MOS scale. Listeners express their opinion on the quality of the perceived speech signal (no reference presented)

Excellent	5
Good	4
Fair	3
Poor	2
Bad	1

Table 5.2 Grades in the *detectability opinion* scale. Listeners give their opinion on the detectability of some property of a sound

Objectionable	3
Detectable but not objectionable	2
Not detectable	1

Table 5.3 Grades in the DMOS scale. Listeners are asked to describe *degradation* in the second signal in relation to the first signal

Inaudible	5
Audible but not annoying	4
Slightly annoying	3
Annoying	2
Very annoying	1

with three samples: A, B, and X. The listener is asked to select which of the samples A and B is identical to X. This method provides a simple assessment of listener reliability, through the calculation of the number of correct answers.

A standardized extension of the ABX method is described in [5.5]. The subject assesses one of three stimuli: A, B, and X. The known reference is always available as stimulus X. The hidden reference and the object are randomly assigned to A and B. The test subject is asked to assess the impairments in A compared to X, and B compared to X, according to a continuous impairment scale. [5.5] defines methods for the subjective assessment of *small impairments* in audio signals (aimed at high-quality audio signals). This method is aimed at so-called expert listeners (who have been trained). The higher the quality reached by the systems to be tested, the more important it is to have expert listeners.

A test procedure more suitable for the subjective assessment of *intermediate*-quality audio signals is the *multistimulus test with hidden reference and anchor* (MUSHRA), standardized in [5.6]. In this test, both a known reference signal and hidden anchors are used. During the test, the users can switch at will between the known reference signal (which is not scored and is known to be the original) and any of the signals under test. They are required to score the stimuli on a continuous quality scale according to a continuous quality scale from 0 to 100, using sliders on an electronic display. Thus, the subjects score the stimuli according to a *continuous quality scale*. This scale is also extensively used for the evaluation of video quality [5.7]. Continuous quality scales provide a continuous rating system to avoid quantization errors, but they are divided into five equal lengths that correspond to the conventional ITU-R fine-point quality scale. Results obtained from continuous quality methods should not be treated as absolute scores but as difference scores between the known reference and the test condition.

In the DCR methods, listeners always rate the amount by which the processed (second) sample is degraded relative to the unprocessed (first) sample. *Comparison category rating* (CCR) procedures can be seen as a refinement of DCR tests. A CCR procedure is a comparison test in which the listeners identify the quality of the second stimulus relative to the first using a two-sided rating scale. This scale is referred to as *comparison mean opinion score* (CMOS) and is presented in Table 5.4. The key idea in the CCR test method is to eliminate the ordering restriction of the DCR test at the expense of doubling the total number of trials. For half of the trials (chosen at random), the unprocessed speech is presented first, followed by the processed speech. For the other half, the order is reversed. An advantage of the CCR method over the DCR method is the possibility to assess speech processing that either degrades or improves the quality of the speech.

Table 5.4 Grades in the CMOS scale. Listeners grade the perceived quality of a speech signal in relation to a reference speech signal

Much better	3
Better	2
Slightly better	1
About the same	0
Slightly worse	-1
Worse	-2
Much worse	-3

For completeness, we mention a method for the general assessment of sound quality that is described in [5.8]. For this method, expert listeners are always preferred over non-expert listeners. Based on the nature and the purpose of the test, each of the scales presented in Tables 5.1, 5.3, and 5.4 can be used.

Conversational Quality

So far we have discussed test procedures concerned with the *listening quality*. In these tests, the listener rates the signal that is received from the far end. Effects such as echoes at the talker side and transmission delays are ignored. *Conversational* quality refers to how listeners rate their ability to converse during the call (which includes listening quality impairments).

In conversational tests, a pool of subjects are placed in interactive communication scenarios, and asked to complete a task [5.3]. At the end of the conversation, the listeners give an opinion on the connection on five-point category-judgment scale (Table 5.1). The test subjects also give their binary response on a *difficulty scale*. The test subjects answer the question: *Did you or your partner have any difficulty in talking or hearing over the connection* with {Yes = 1, No = 0}. Conversational quality tests are standardized in [5.3], and a thorough study can be found in [5.9].

Conversational tests are exclusively not reference based, yet they are significantly more complex to design and control than conventional listening tests. It is therefore beneficial to perform listening tests where possible. Research on the relations between listening and conversational quality can be found in [5.10].

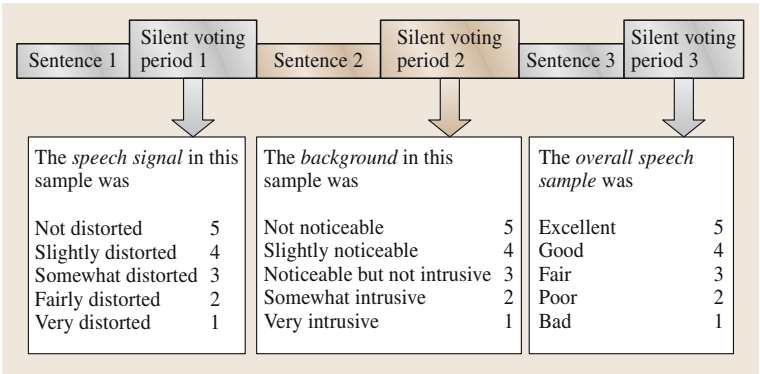


Fig. 5.1 The ITU-T P.835's scheme for evaluating the subjective quality of noise-suppression algorithms. Each test sample is comprised of three subsamples, where each subsample is followed by a silent voting period

5.2.2 Multidimensional Metric (Diagnostic Speech-Quality)

In many cases, the designers of a speech-processing algorithm want to obtain information about speech quality together with diagnostic information. This task requires a *multidimensional metric* subjective test, which provides more insight into system faults and shortcomings than tests providing only an overall speech quality measure.

A procedure that assesses the speech quality on a multidimensional metric is the *diagnostic acceptability measure* (DAM) [5.11], which provides more-systematic feedback and evaluates speech quality on 16 scales. These scales belong to one of three categories: signal quality, background quality, and overall quality. A weighted average of all these scales forms a composite measure that describes the condition under test. The DAM procedure is designed for trained (experienced) listeners and, in contrast to most other measures, the speech material is known to the listeners.

While the DAM procedure is general purpose, it is also possible to design *specialized* multidimensional metrics. One example is the procedure standardized in [5.12], which describes a methodology for evaluating the subjective quality of speech in noise and noise-suppression algorithms. The methodology uses separate rating scales to estimate independently the subjective quality of the speech signal alone, the background noise alone, and overall quality. Each speech sample is comprised of three subsamples, where each subsample is followed by a silent voting period Fig. 5.1. Recent experiments with this procedure are presented in [5.13].

Table 5.5 Grades in the listening effort scale. Listeners assess the *effort required* to understand the meaning of sentence

Complete relaxation possible; no effort required	5
Attenuation necessary; no appreciable effort required	4
Moderate effort required	3
Considerable effort required	2
No meaning understood with any feasible effort	1

5.2.3 Assessment of Specific Quality Dimensions

The *diagnostic rhyme test* (DRT) [5.14] evaluates speech intelligibility. It uses a set of isolated words to test for consonant intelligibility in the initial position. The DRT uses 96 pairs of confusable words (which differ by a single acoustic feature in the initial consonant) spoken in isolation. First, the subject is presented visually with a pair of rhymed words. Then, one word of the pair (selected at random) is presented aurally and the subject is required to indicate which of the two words was played.

The *modified rhyme test* (MRT) [5.15, 16] is an extension to the DRT. It evaluates the intelligibility of both initial and final consonants. A set of six words is played one at a time and the listener marks which word they think they hear.

Intelligibility tests are rarely used to assess the quality of speech coding systems, since while most systems affect naturalness, they do not degrade speech intelligibility significantly. In other applications, such as speech synthesis, intelligibility tests are the major testing tool. In addition to DRT and MRT, other popular tests are the Bellcore test [5.17], and the minimal pairs intelligibility (MPI) test [5.18].

A frequently used opinion scale that focuses on the listener's ability to understand the meaning of the sentence is the *listening effort* scale [5.3] (Table 5.5). Methods using the listening effort scale generally yield results that are better correlated with conversational

opinion scores than methods using the listening quality scale [5.19].

5.2.4 Test Implementation

The designer of a subjective test should take care of:

1. the selection of the test material
2. the selection of the test subjects
3. the choice of the test procedure
4. the analysis of the test results

Careful experimental design and planning is needed to ensure that uncontrolled factors, which can cause ambiguity in the test results, are minimized.

Speech material should consist of simple, short, meaningful, and preferably phonetically balanced [5.20], sentences. In the ACR test, the sentences should be made into sets of two without an obvious connection of meaning between the sentences in a set (Fig. 5.2). If the impairments are simulated, the test designer has the responsibility to make the model as close to reality as possible. As an example, for high noise levels people change their talking behavior (the Lombard effect [5.21]), which is often not considered when noise recording is digitally added to the clean-speech database.

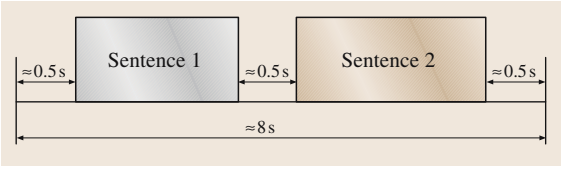


Fig. 5.2 A typical MOS test uses stimuli that contain two short sentences, separated by a 0.5 s seconds silence gap. The resulting two-utterance stimulus has a duration of about 8–10 s

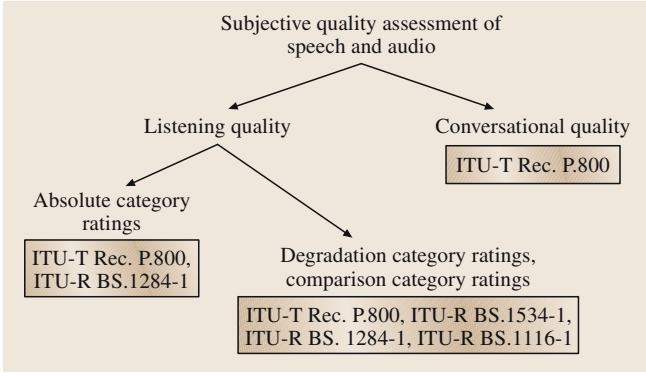


Fig. 5.3 Classification of subjective quality assessment methods and related ITU standards and recommendations

The most important requirements for the selection of the test subjects are:

1. they must be native speakers of the language in which the tests are being conducted,
2. they do not have known hearing defects, and
3. they have not recently been involved in quality assessment tests of the same or similar speech processing system.

Note that some test procedures require experienced listeners, e.g., [5.5], while others require naïve listeners, e.g., [5.12]. In general, a listening test should use experienced listeners when impairments are small, or when fast test convergency is needed (i. e., small number of listeners). However, listening tests using naïve listeners represent the actual conditions under which the communication system will be used better.

A classification of the most popular subjective tests standardized by the ITU is shown in Fig. 5.3. As discussed earlier, the choice of a particular test procedure is dependent on the type of impairments and the required format (e.g., overall quality, diagnostic information, etc.). Different test procedure may require different listening conditions (headphones or loudspeakers, scoring on a computer or voting on paper, etc.), but these conditions cannot be changed within the test. For most of the test procedures it is recommended that test subjects are given a break after 20–30 min of listening. It is vital that subjects are not overloaded to the point of decreased accuracy of judgment.

Test designers usually include *reference conditions* (well-defined test conditions) in the quality assessment procedure. There are two major reasons to introduce reference conditions in listening tests. The first is to provide a convenient means for comparing subjective test results from different laboratories. The second is that reference conditions provide a spread in quality level, which increases the consistency of human ratings across tests.

Reference conditions, used in telecommunications, typically include a best possible condition (the original signal or a high-quality speech codec), as well as conditions where controlled degradation is introduced. One of the most commonly used types of reference signals is the *modulated noise reference unit* (MNRU), which is standardized in [5.22]. MNRU produces random noise with an amplitude that is proportional to the instantaneous speech amplitude (i. e., multiplicative noise):

$$y_k = s_k\left(1+10^{-Q/20}v_k\right), \tag{5.1}$$

where y_k is the output noise-modulated speech, s_k is the original speech signal, v_k is the random noise, k is the sample index, and Q is the ratio of the speech power to the modulated noise power.

The average of the opinion scores should be calculated for each test condition. A variety of statistical procedures is available to compare the results obtained from listening tests.

The t-test is a commonly used method to identify a significant difference in means between two conditions. The first step is to calculate the minimum significant difference (MSD) threshold (at the 95% significance level):

$$\mathrm{MSD} = t_{95\%,2(M-1)}\sqrt{\frac{\sigma_\mathrm{A}^2+\sigma_\mathrm{B}^2}{M}}\,, \tag{5.2}$$

where M is the number of observations, σ_A^2 is the variance of the scores for condition A, σ_B^2 is the variance of the scores for condition B, and $t_{95\%,2(M-1)}$ is a tabulated critical value from Student's T distribution at the 95% significance level with $2(M-1)$ degrees of freedom.

The second step is to compare the absolute difference between the mean values for the two conditions $|\mu_\mathrm{A}-\mu_\mathrm{B}|$ with the previously calculated MSD threshold. If $|\mu_\mathrm{A}-\mu_\mathrm{B}| > \mathrm{MSD}$ one can conclude that the quality of condition A is significantly different from the quality of condition B at the 95% significance level.

If there is a need to compare more than two conditions, or compare conditions created by more than one independent variable, one has to consider the usage of the analysis of variance (ANOVA) or honestly significant difference (HSD) tests [5.23].

5.2.5 Discussion of Subjective Tests

Since the auditory event that causes the perception of quality degradation is not available for direct measurement, the test subjects describe their perception on quality scales, which may be of different types. In a *category scale*, the subject assigns a certain class (usually labeled with numbers or symbols) to their perception of signal quality. In an *ordinal scale* the subject arranges test samples into an order (e.g., the loudest sample first). In *interval* and *ratio scales* the differences between classes, or correspondingly the ratio between classes, is quantified.

In ACR procedures, the test subjects rate each presented test item on a discrete ordinal scale, for example labeled as described in Table 5.1. Each of these categories is assigned a number, and the total average of all rating results is often expressed as an MOS. In ACR procedures, the mean values are calculated assuming that each category occupies the same interval on a perceptual continuum, and the statistical processing of the data assumes that this five-point ITU scale is an interval scale.

Some studies [5.24] indicate that the intervals represented by the quality scale labels are not equal. There are also indications that the scale labels cannot be translated adequately into different languages, such that the scale is *equal* in different countries. It is common to apply statistical tests (e.g., analysis of variance) to recorded scores. In the cases where MOS scores are not presumed to represent a linear scale, statistics for ordinal scales may need to be applied [5.25].

In the quality assessment procedures discussed so far, we have assumed that the speech quality does not vary significantly during the speech sample being evaluated. In reality, quality of service may vary even during a single conversation and attention must be paid when investigating time-varying impairments. Time-varying speech quality has been the main focus of several studies, e.g., [5.26–32]. Some important observations from this research are that

1. the long-term perceived speech quality scores are lower than the time average of the corresponding short-term perceived speech quality scores,
2. the perceived long-term quality decreases when the variance of the short-term speech quality increases,
3. listeners detect decreases in speech quality more quickly than increases in speech quality, and
4. long-term scores are more strongly influenced by events near the scoring time than by earlier events.

A methodology for the assessment of time-varying speech quality is standardized in [5.33]. The test subjects are asked to assess the speech quality continuously by moving a slider along a continuous scale so that its position reflects their opinion on quality at that instant (the slider position should be recorded every 500 ms). For each utterance a *mean instantaneous judgment* is obtained by averaging individual instantaneous judgments over the subjects. At the end of the utterance, subjects are asked to rate its overall quality on a five-point ACR scale. For each utterance a *mean overall judgment* is obtained by averaging individual overall judgments over the subjects on the ACR scale. A substantial difference between the continuous score and the overall score indicates a time-varying quality level.

5.3 Objective Measures

In this Section we discuss *objective* quality measures: computer algorithms designed to estimate quality degradation. Subjective tests are believed to give the *true* speech quality. However, the involvement of human listeners makes them expensive and time consuming. Subjective tests are not suitable to monitor the quality of service (QoS) of a network on a daily basis, but objective measures can be used for this purpose at a very low cost. The main aspects that affect the applicability of objective and subjective measures are summarized in Table 5.6.

Similarly to the subjective tests, some objective quality measures are designed to assess listening quality, while others assess conversational quality. Alternatively, the classification of objective quality measures can be based on the type of input information they require: intrusive quality measures require access to both the original and distorted speech signal, while non-intrusive measures base their estimate only on the distorted signal, as illustrated in Fig. 5.4.

Early work on objective quality assessment focused exclusively on intrusive methods, and non-intrusive methods have received attention only in the last decade. If only the distorted signal is available, sophisticated modeling of the speech and/or distortions is typically needed. In contrast, intrusive measures range from very simple to sophisticated models that consider human perception.

5.3.1 Intrusive Listening Quality Measures

Simple Time- and Frequency-Domain Measures

The simplest class of intrusive objective quality measures consists of waveform-comparison algorithms, such as those based on the signal-to-noise ratio (SNR) and segmental SNR (SSNR). These two algorithms are easy to implement, have low computational complexity, and can provide indications of perceived speech quality for a specific waveform-preserving speech system [5.34]. Unfortunately, when used to evaluate coding and transmission systems in a more-general context, SNR and SSNR show little correlation to perceived speech quality. These measures are also sensitive to a time shift, and therefore require precise signal alignment, which is not always a trivial problem.

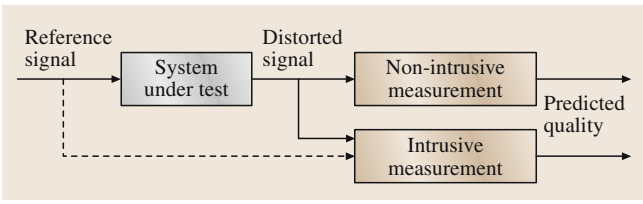


Fig. 5.4 Intrusive and non-intrusive types of quality assessment. Non-intrusive algorithms do not have access to the reference signal

Table 5.6 Comparison of subjective and objective methods for quality assessment. The symbol "+" is used to denote that the method is advantageous over the other method, denoted by "−"

	Subjective measures	Objective measures
Cost	−	+
Reproducibility	−	+
Automation	−	+
Unforeseen impairments	+	−

The overall SNR distortion measure between an original $\boldsymbol{s}$ and a distorted $\boldsymbol{y}$ speech vector is calculated as [5.34]:

$$d_{\mathrm{SNR}}(\boldsymbol{s},\boldsymbol{y}) = 10\log_{10}\left(\frac{\boldsymbol{s}^{\mathrm{T}}\boldsymbol{s}}{\boldsymbol{e}^{\mathrm{T}}\boldsymbol{e}}\right) , \tag{5.3}$$

where $\boldsymbol{e} = \boldsymbol{s} - \boldsymbol{y}$. The vector dimension is sufficient to contain the entire utterance.

The SSNR is calculated by splitting the two vectors into smaller blocks and calculating an SNR value for each of these blocks. The final SSNR value is obtained by averaging the per-block SNR values, e.g., [5.35]:

$$d_{\mathrm{SSNR}}(\boldsymbol{s},\boldsymbol{y}) = \frac{1}{N}\sum_{n=1}^{N} 10\log_{10}\left(\frac{\boldsymbol{s}_n^{\mathrm{T}}\boldsymbol{s}_n}{\boldsymbol{e}_n^{\mathrm{T}}\boldsymbol{e}_n}\right) , \tag{5.4}$$

where N is the total block number, n is the block index, and the per-block error vector is defined as $\boldsymbol{e}_n = \boldsymbol{s}_n - \boldsymbol{y}_n$. A typical block length is 5 ms. In a perceptual modification of this measure, studied in [5.36], weights that depend on the time-varying spectral envelope of the original speech are applied.

Frequency-domain measures are known to be significantly better correlated with human perception [5.34], but still relatively simple to implement. One of their critical advantages is that they are less sensitive to signal misalignment. Some of the most popular frequency-domain techniques are the *Itakura–Saito* (IS), the cepstral distance (CD), the log-likelihood (LL), and the log-area-ratio (LAR) measures [5.37].

The gain-normalized spectral distortion (SD) is widely accepted as a quality measure of coded speech

spectra [5.38]. In its most commonly used form, the SD evaluates the similarity of two autoregressive envelopes on a per-frame basis:

$$d_{\mathrm{SD}}^{(n)}(\boldsymbol{s},\boldsymbol{y}) = \int_{-\pi}^{\pi} \left[10\log_{10}\left(\frac{P_s(\omega,n)}{P_y(\omega,n)}\right)\right]^2 \frac{\mathrm{d}\omega}{2\pi}\,, \quad (5.5)$$

where $P_s(\omega,n)$ and $P_y(\omega,n)$ are the autoregressive spectra of the clean and processed signal. The per-frame distances are generally combined into a global (per-signal) distortion in the form of a root-mean SD:

$$d_{\mathrm{SD}}(\boldsymbol{s},\boldsymbol{y}) = \frac{1}{N}\sqrt{\sum_{n=1}^{N} d_{\mathrm{SD}}^{(n)}(\boldsymbol{s},\boldsymbol{y})}\,, \quad (5.6)$$

where N is the total number of frames. A further enhancement of SD is proposed in [5.39], where the authors apply weighting with a perceived loudness function, which takes in consideration frequency-dependent perception sensitivities. It is noted that perceived distortion of the spectral fine structure is not considered in the SD measures described above.

The distinguishing characteristic of both waveform comparison and frequency-domain techniques is that the basic measure operates on a per-frame basis and that they use simple schemes to combine the estimated per-frame distortions [5.40]. The most commonly employed method for the construction of a global objective distortion measure over a number of N frames can be computed by arithmetically averaging the per-frame computed distance measures. In a more-sophisticated form of this basic measure, unequal contributions to perception from each speech frame can be taken into account. An unequal distribution can be related to frame energy and/or voicing.

The measures discussed in this subsection only have meaning when applied to frames where speech is present. A known problem is that authors do not specify, or use different rules, to select the subset of N speech active frames from the entire frames set. To obtain repeatable results, it is advisable to use speech activity detection based on a speech level meter [5.41].

The intrusive quality measures methods discussed thus far are based on simple and tractable mathematical models. The next topic in our discussion covers a more-sophisticated family of quality measures that is based on knowledge of the human auditory system.

Psychoacoustically Motivated Measures

Many intrusive quality measures are based on mimicry of the human auditory system. This approach has led to highly accurate objective performance measures that are useful in many contexts where the original signal is present.

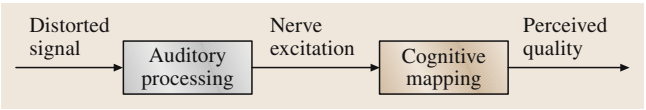


Fig. 5.5 Human perception of speech quality, involving both hearing and judgment

The process of human assessment of speech quality can be described as consisting of two stages, as illustrated in Fig. 5.5. In the first auditory processing (hearing) stage, the received speech acoustic signal is transformed into an auditory nerve excitation. Essential elements of auditory processing include bark-scale frequency warping and spectral power to subjective loudness conversion. The second stage of the quality assessment process entails cognitive processing in the brain, where compact features (that contain information about the anomalies in the speech signal) are extracted from auditory excitations. These features are combined to form a final impression of the perceived speech signal quality. The cognitive models of speech distortions are less well developed than the auditory model.

Figure 5.6 shows an outline of an ideal mimicry-based speech quality assessment algorithm that incorporates both stages of subjective quality assessment. Both original and distorted signals are first subjected to a perceptual transform that mimics the auditory periphery. Then a process mimicking high-level cognitive process extracts patterns related to the language specifics, context, etc. Finally, the distance between the expected (original) and actual patterns is mapped to a selected speech quality scale. Unfortunately, this scheme is currently not realizable, since the cognitive processing performed by the human brain is largely unknown.

The human auditory periphery is well understood and perceptual transforms are thought to be reasonably accurate. The transforms result in a signal representation that is scaled such that upon thresholding (representing a minimum precision) only perceptually relevant infor-

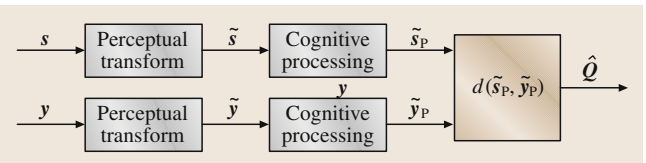


Fig. 5.6 A speech quality assessment algorithm that mimics the process of human quality assessment

mation is retained. The transforms reflect the fact that the resolution of the ear is not uniform on the Hertz scale and that the relation between perceived loudness and signal intensity is nonlinear. The sensitivity of the ear is a function of frequency and the corresponding absolute hearing thresholds have been characterized. Finally, many studies have demonstrated time- and frequency-domain masking effects [5.42–45]. A study presented in [5.46] argues that nonuniform frequency resolution and nonlinear loudness perception are the most important properties of auditory models.

Historically, researchers have followed two different paths to incorporate knowledge of the auditory periphery into quality assessment schemes. The first is based on the *masking threshold concept*, where the reference signal is used to calculate an estimate of the actual masking threshold operating on a short-term frequency spectrum. The difference between the reference and processed signals is evaluated relative to this masking threshold [5.36, 47]. The second approach aims to compare *internal representations* directly. Based on a model of the auditory periphery, an internal representation of both signals is calculated. This internal representation contains the information that is available to the brain for comparison of signals [5.48]. Note that the second scheme is closer to the desired scheme shown in Fig. 5.6.

It can be shown that an appropriate weighting of the difference for the masking approach is precisely the inverse of the masking threshold. The frequency spectra weighted by the inverse masking function can then be interpreted as a simple internal representation of the human ear, reconciling the two historic approaches.

Internal representations can have different levels of sophistication. The bark spectral distortion (BSD) [5.49] can be seen as one of the first and simplest internal measures. The BSD is the averaged Euclidean distance between the original and distorted speech signals in the Bark domain. A similar measure is the information index (II) [5.50], according to which the auditory system is modeled by dividing the spectrum into 16 critical bands and applying empirical frequency weights and a hearing threshold for each band. The coherence function (CF) [5.51] is a measure of the signal-to-distortion ratio. The objective of the coherence function is to turn off uncorrelated signals and pass correlated signals.

A more-sophisticated internal presentation is used in [5.52], which describes a perceived speech quality assessment algorithm called perceptual speech quality measure (PSQM). Its scope is limited to higher-bitrate speech codecs operating over error-free channels. The transformation from the physical to the psychophysical (internal) domain is performed in three steps – frequency warping, time–frequency smearing, and level compression [5.52].

In ITU-R. BS.1387-1, perceptual quality assessment for digital audio (PEAQ) [5.53, 54], concurrent frames of the original and processed signals are each transformed to a basilar membrane representation (the internal representation), and differences are further analyzed as a function of frequency and time by a cognitive model.

ITU-T Rec. P.862, perceptual evaluation of speech quality (PESQ) [5.55], measures the effects of one-way speech distortion. The effects of loudness loss, delay, sidetone, echo, and other impairments related to two-way interaction are not reflected in PESQ scores. Factors for which PESQ has demonstrated acceptable accuracy are: speech input levels to a codec, transmission channel errors, packet loss and packet loss concealment with CELP codecs, bitrates if a codec has more than one bitrate mode, transcodings, environmental noise at the sending side, the effects of varying delay in listening only tests, short-term time warping of audio signal, and long-term time warping of audio signal.

ITU-T Rec. P.862 was designed to evaluate narrowband (3.4 kHz) speech quality, and cannot deal with wideband (7 kHz) speech quality. A recent research focus has been the development of a wideband extension for PESQ, ITU-T Rec. P.862.2 [5.56].

The above discussion centered on processing that mimics the human auditory periphery. The outcome is a representation that corresponds to that in the auditory nerve. This representation must then be mapped into a quality measure. It is important to note that the lack of knowledge about high-level cognitive brain processing cannot be compensated for by more-sophisticated models of the auditory periphery. Less-audible parts of the signal may be more objectionable if they are of higher importance for the pattern extraction and comparison process performed by the human brain.

Some important cognitive effects that affect the process of quality judgment are:

1. linear distortion is generally less objectionable than nonlinear distortion,
2. speech-correlated distortion is more objectionable than uncorrelated distortion,
3. if the local *information* complexity is high, then distortion is less objectionable, and
4. distortion in some spectral–temporal components, such as formants, is more objectionable, since they carry more information.

Even though not completely understood, these effects have been used successfully by researchers to improve perceived quality in speech coding [5.57, 58], and enhancement [5.59] applications. These studies are motivated by the evidence that noise is aurally masked by rapid changes in the speech signal [5.43].

Perhaps the simplest way to develop a relationship between two internal representations corresponding to auditory nerve signals and a target subjective opinion scale is to carry out a weighted summation of their difference along the frequency and time axes. As an example, PESQ performs integration in three steps, first over frequency, then over short time intervals, and finally over the entire speech signal. The authors of [5.60–62] present an error integration scheme that is more consistent with high-level brain processing. *Holier* et al. [5.60] uses an entropy measure of the difference internal representation. The measuring normalizing blocks (MNB) algorithm [5.61, 62] utilizes a relatively simple perceptual transform, but a sophisticated error pooling system. MNB uses a hierarchical structure of integration over a range of time and frequency intervals.

A conceptually simple objective speech quality measure can also be obtained by integrating elements of high-level brain processing. An interesting measure based on distortion in the spectral peaks of speech was proposed in [5.63]. The approach proposed in [5.64] is to measure the phonetic distance between the original and distorted signals (calculated as the weighted difference between spectral slopes over several frequency bands).

A more-general approach to simulate this process is through statistical mapping and data mining [e.g., Gaussian mixture models (GMMs) or neural networks (NNs)], or clustering [5.65]. Another example can be found in [5.66], where the authors recognize the importance of the high-level cognitive process and apply a statistical data-mining approach. In the approach of [5.66], a large pool of candidate features is created and the ones that lead to the most accurate prediction of perceived quality are selected.

5.3.2 Non-Intrusive Listening Quality Measures

Intrusive objective speech quality measures can provide a performance measure for a communication system without the need for human listeners. However, intrusive measures require the presence of the original signal, which is not available in some important applications, including QoS monitoring of telecommunication networks. For such applications *non-intrusive* quality assessment must be used.

A wide variety of approaches has been used to obtain non-intrusive quality assessment. These include methods that assess the possibility of the signal being produced by human physiology, methods that compare to the nearest signal from a speech database, and methods that learn the human mapping between signal and quality directly.

Reference [5.67] reports a non-intrusive speech quality assessment that attempts to predict the likelihood that the signal has been generated by the human vocal production system. To achieve this, the speech signal under assessment is first reduced to a set of features. This parameterized data is then used to estimate the perceived quality by means of physiologically based rules.

The measure proposed in [5.68] is based on comparing the output speech to an artificial reference signal that is appropriately selected from an optimally clustered codebook. The perceptual linear prediction (PLP) [5.69] coefficients are used as a parametric representation of the speech signal. A fifth-order all-pole model is performed to suppress speaker-dependent details of the auditory spectrum. The average distance between the unknown test vector and the nearest reference centroids provides an indication of speech degradation.

The authors of [5.70, 71] propose a method that employs intrusive algorithms. However, they avoid the need for the original signal. The method is based on measuring packet degradations at the receive end. The measured degradation is applied to a *typical* speech signal to produce a signal that is similarly degraded. An intrusive algorithm can then be used to map the speech signal and degradation signal to speech quality.

A novel, perceptually motivated, speech quality assessment algorithm based on the temporal envelope representation of speech is presented in [5.72] and [5.73].

A non-intrusive speech quality assessment system, based on a speech spectrogram, is presented in [5.74]. An interesting concept in this approach is that accurate estimation of speech quality is achieved without a perceptual transform of the signal.

The ITU standard for non-intrusive quality assessment is ITU-T P.563 [5.75]. It consists of a combination of a number of approaches. A total of 51 speech features are extracted from the signal. *Key features* are used to determine a dominant distortion class, in each of which a linear combination of features is used to predict a so-called intermediate speech quality. The final speech

quality is estimated from the intermediate quality and 11 additional features.

Another approach is to use perceptually motivated spectral envelope representations in combination with a mapping from this representation to the quality measure through a GMM [5.76,77]. The authors of [5.68,78] used a similar approach but with mappings based on hidden Markov models (HMMs) and neural networks.

A recent development [5.79] simplifies this method and combines the GMM mapping with a small set of features selected for optimal performance from a large set of features. As a result, the algorithm has very low complexity, requiring only a small fraction of the computational capability of a mobile phone.

5.3.3 Objective Measures for Assessment of Conversational Quality

The emphasis in this chapter, and also in the literature on quality assessment, is on listening quality. However, some applications may require the assessment of conversational quality, i. e., to include impairments such as delay and talker echo.

The E-model is a tool for predicting how an *average user* would rate the voice quality of a phone call with known characterizing transmission parameters (currently 21 input parameters). The E-model is standardized in ITU-T Rec. G.107 [5.80]. The objective of the E-model is to determine a transmission quality rating, i. e., the R factor, with a typical range between 0 and 120. The R factor can be converted to estimated listening and conversational quality MOS scores. The E-model does not compare the original and received signals directly. Instead, it uses the sum of equipment impairment factors, each one quantifying the distortion due to a particular factor. Impairment factors include the type of speech codec, echo, averaged packet delay, packet delay variation, and the fraction of packets dropped. A fundamental assumption is that the impairments on the psychological scale are additive [5.81]. The transmission quality rating is determined as

$$R = R_0 - I_s - I_d - I_{e\text{-}eff} + A \,, \tag{5.7}$$

where R_0 represent the basic signal-to-noise ratio (noisiness), including noise sources such as circuit and room noise, I_s is a combination of all impairments that occur simultaneously with the voice signal (loudness), I_d represents the impairments caused by the delay (echo and delay), $I_{e\text{-}eff}$ represents impairments caused by low-bitrate codecs and packet losses (distortion), and A allows for compensation of impairments factors.

The broader scope of conversational quality assessment, as compared to listening quality assessment, is illustrated in Fig. 5.7. For completeness, we note that speech quality can also by discussed solely from the viewpoint of the context of the talker [5.82], with echo and sidetone distortion being the main impairments. There are also studies on the possibility of decomposing conversational quality into listening, talking, and interaction quality, and building a prediction from these components.

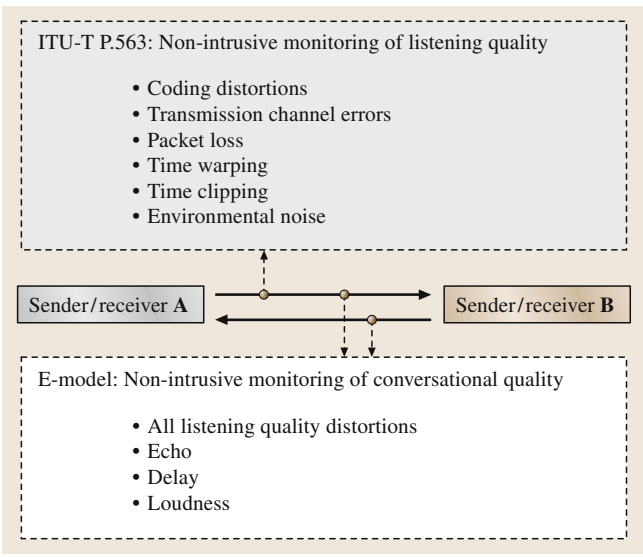


Fig. 5.7 Non-intrusive monitoring of listening and conversational quality over the network

5.3.4 Discussion of Objective Measures

Objective measures of speech quality are now relatively mature, as is reflected in the standards defined by the ITU-T. These standards and a corresponding classification of the methods are presented in Fig. 5.8. This standardization indicates that the performance of these measures is satisfactory for many practical applications. In this subsection we discuss how the performance of objective quality measures can be evaluated, and provide an indication of the performance of the state-of-the-art algorithms.

The development of a computer algorithm that predicts the output of subjective quality test well is not straightforward. Complications come from the fact that speech perception is influenced by grammar rules and semantic context. Modeling of factors such as the speaker's emotional state and expectations of speech quality

is even more difficult. It is therefore critical that newly developed objective quality algorithms are properly *calibrated* to the output of subjective quality tests. Objective measures that generate results that closely approximate subjective test results are more useful than objective measures that show larger deviation. To facilitate the reproducibility of the evaluation process, it has to be trained over a large multilanguage database that contains a wide range of distortions, e.g., [5.83].

In the literature, two methods are commonly used to evaluate the performance of quality assessment algorithms. The first is suitable for codec or transmission equipment evaluation. The subjective MOS for the speech files within a particular test condition are first averaged together. The objective MOS are likewise grouped and averaged. Then, the correlation coefficient R and the root-mean-square error (RMSE) ε between the per-condition averaged subjective and the objective MOS over all the conditions in the database are calculated. Let the measured subjective quality be denoted by Q, and the predicted objective quality by $\hat{Q}$, then the RMSE is given by

$$\varepsilon = \sqrt{\frac{\sum_{i=1}^{L}(Q_i - \hat{Q}_i)^2}{L}} \tag{5.8}$$

and the correlation coefficient is defined as

$$R = \frac{\sum_{i=1}^{L}(\hat{Q}_i - \mu_{\hat{Q}})(Q_i - \mu_Q)}{\sqrt{\sum_{i=1}^{L}(\hat{Q}_i - \mu_{\hat{Q}})^2}\sqrt{\sum_{i=1}^{L}(Q_i - \mu_Q)^2}}\,, \tag{5.9}$$

where μ_Q and $\mu_{\hat{Q}}$ are the mean values of the introduced variables and L is the number of conditions in the database.

An alternative method to evaluate the accuracy of objective quality algorithms is to calculate the correlation and RMSE between the objective and subjective MOS for each utterance. This concept is better suited for the assessment of performance in voice quality monitoring applications. The RMSE and correlation coefficient are calculated in a similar fashion as described above,

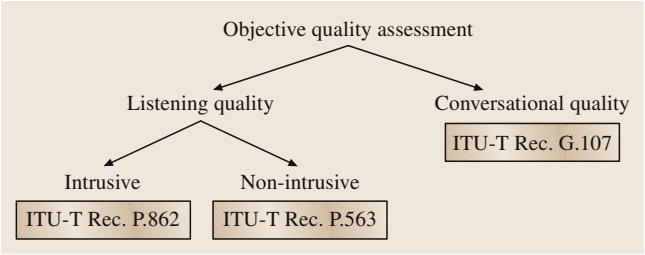


Fig. 5.8 Classification of objective quality assessment methods and related ITU standards and recommendations

but the summation is over MOS-labeled utterances in the database, and L is the number of these utterances.

One typically overlooked problem in the training and evaluation of objective quality algorithms is that most of the currently available data are for ACR-type listening tests. Thus, intrusive algorithms (e.g., [5.52, 84]) were designed to predict listening quality not based on DCR testing and with a DMOS scale, as is natural, but in terms of MOS values. To the best of our knowledge, there are no studies to assess the impact of this mismatch. This issue is not relevant to non-intrusive algorithms such as ITU-T P.563, [5.75], which are designed to predict the outcome of ACR subjective tests.

Both intrusive and nonintrusive algorithms provide a performance that is useful for practical applications. The intrusive PESQ algorithm (ITU-T Rec. P.862.2) has been observed to have a correlation coefficient R of 0.85–0.95 [5.85, 86]. The nonintrusive ITU-T P.563 standard has a correlation coefficient of 0.8–0.9 [5.79, 86]. Recent work describing new nonintrusive measures reports even higher correlations [5.79].

The discussed measures of listening and conversational quality are designed to predict the speech quality from the simultaneous effect of a large number of distortions. An objective quality assessment measure can also be designed to operate in a particular environment only (e.g., specific speech coding standards in the context of a particular mobile network). These constraints can be used to obtain higher system accuracy and reduce complexity and memory requirements [5.87].

5.4 Conclusions

Perhaps the first question that comes to mind after reading an overview on speech quality assessment is: why is it so difficult to assess speech quality and why are so many scales and test procedures used? The explanation is that the auditory event that causes the perception of quality degradation is always

Table 5.7 A survey of typical quality assessment problems and recommended test methodology

Assessment problem	*Quality assessment method*
Subjective assessment of overall conversational quality	Absolute category rating (ACR), Conversation difficulty scale [5.3]
Multidimensional subjective speech quality assessment	Diagnostic acceptability measure (DAM) [5.11]
Subjective assessment of speech intelligibility	Diagnostic rhyme test (DRT) [5.14], Modified rhyme test (MRT) [5.15],
Subjective assessment of general audio quality	ITU-R BS.1284-1 [5.8]
Subjective assessment of high audio quality	ITU-R BS.1116-1 [5.5]
Subjective assessment of intermediate audio quality	MUSHRA [5.6]
Subjective assessment of overall listening quality of speech processing system	Absolute category rating (ACR) [5.3]
Subjective evaluation of speech processing algorithms of similar quality (typically for high-quality speech processing algorithms)	Degradation category rating (DCR) [5.3]
Subjective evaluation of systems that may increase or decrease the quality of the input speech	Comparison category rating (CCR) [5.3]
Subjective evaluation of threshold values of certain quantities	Quantal-response delectability [5.3]
Subjective evaluation of noise suppression algorithm	ITU-T Rec. P.835 [5.12], Comparison category rating (CCR) [5.3]
Objective assessment of listening speech quality (reference signal not available)	ITU-T Rec. P.563 [5.75]
Objective assessment of listening speech quality (reference signal available)	PESQ [5.55]
Objective assessment of audio quality (reference signal available)	PEAQ [5.54]
Objective assessment of conversational speech quality (network parameters available)	E-model [5.80]

unknown and can be only projected on a particular scale(s) through the response of the test subject. Thus, we generally deal with one or more projections of speech quality, suitable for the particular application.

It is difficult to give general guidance on which assessment method is adequate for a specific quality assessment problem. Nevertheless, in Table 5.7 we make an attempt to relate some of the typical quality assessment situations with an adequate test methodology.

In line with the topic of this book, we did not cover general audiovisual quality testing in this chapter. Readers interested in this related topic are referred to ITU-T P.920 *Interactive test methods for audiovisual communications* [5.88]. In this recommendation, a five-point quality scale is suggested for assessing video, audio, and overall quality. Trends in speech quality assessment can be obtained from ITU-T study group 12 [5.89], which is the lead study group at the ITU-T on network performance and QoS.

References

5.1 ITU-T Rec. G.113: *Transmission impairments* (Geneva 2001)

5.2 ITU-T Rec. P.11: *Effects of transmission impairments* (Geneva 1993)

5.3 ITU-T Rec. P.800: *Methods for subjective determination of transmission quality* (Geveva 1996)

5.4 D. Clark: High-resolution subjective testing using a double-blind comparator, J. Audio Eng. Soc. **30**, 330–338 (1982)

5.5 ITU-R Rec. BS.1116-1: *Methods for the subjective assessment of small impairments in audio systems including multichannel sound systems* (Geneva 1997)

5.6 ITU-R Rec. BS.1534-1: *Method for the subjective assessment of intermediate quality level of coding systems* (Geneva 2005)
5.7 ITU-R Rec. BT.500-11: *Method for the subjective assessment of the quality of television pictures* (Geneva 2002)
5.8 ITU-R Rec. BS.1284-1: *General methods for the subjective assessment of sound quality* (Geneva 2003)
5.9 S. Möller: *Assessment and Prediction of Speech Quality in Telecommunications* (Kluwer Academic, Boston 2000)
5.10 M. Gueguin, R. Bouquin-Jeannes, G. Faucon, V. Barriac: Towards an objective model of the conversational speech quality, Proc. IEEE ICASSP **1**, 1229–1232 (2006)
5.11 W. Voiers: Diagnostic acceptability measure for speech communication systems, Proc. IEEE ICASSP **2**, 204–207 (1977)
5.12 ITU-T Rec. P.835: *Subjective test methodology for evaluating speech communication systems that include noise suppression algorithm* (Geneva 2003)
5.13 Y. Hu, P. Loizou: Subjective comparison of speech enhancement algorithms, Proc. IEEE ICASSP **1**, 153–156 (2006)
5.14 W. Voiers: Evaluating processed speech using the diagnostic rhyme test, Speech Technol. **1**, 338–352 (1983)
5.15 A. House, C. Williams, M. Hecker, K. Kryter: Articulation testing methods: Consonant differenctiation with a closed response set, J. Acoust. Soc. Am. **37**, 158–166 (1965)
5.16 M. Goldstein: Classification of methods used for assessment of text-to-speech systems according to the demands placed on the listener, Speech Commun. **16**, 225–244 (1995)
5.17 M. Spiegel, M. Altom, M. Macchi, K. Wallace: Comprehensive assessment of the telephone intelligibility of synthesized and natural speech, Speech Commun. **9**, 279–291 (1990)
5.18 J. van Santen: Perceptual experiments for diagnostic testing of text-to-speech systems, Comput. Speech Lang. **7**, 49–100 (1993)
5.19 ITU-T Rec. P.830: *Subjective performance assessment of telephone-band and wideband digital codecs* (Geneva 1996)
5.20 A. Huggins, R. Nickerson: Speech quality evaluation using phonemic-specific sentences, J. Acoust. Soc. Am. **77**, 1896–1906 (1985)
5.21 H. Lane, B. Tranel: The Lombard sign and the role of hearing in speech, J. Acoust. Soc. Am. **47**, 618–624 (1970)
5.22 ITU-T. Rec. P.810: *Modulated noise reference unit* (Geneva 1996)
5.23 J. Tukey: *The Problem of Multiple Comparisons* (Princeton University, Ditton 1953)
5.24 B. Jones, P. McManus: Graphic scaling of qualitative terms, SMPTE J. **95**, 1166–1171 (1986)
5.25 M. Dahlquist, A. Leijon: Paired-comparison rating of sound quality using MAP parameter estimation for data analysis. In: 1st ISCA Tutorial and Research Workshop on Auditory Quality of Systems (2003)
5.26 S. Voran: A basic experiment on time-varying speech quality. In: Proc. 4th Int. Conf.: Measurement of Speech and Audio Quality in Networks (2005)
5.27 M. Hansen, B. Kollmeier: Continuous assessment of the time-varying speech quality, J. Acoust. Soc. Am. **106**, 2888–2899 (1999)
5.28 L. Gros, N. Chateu: Instantaneous and overall judgements for time-varying speech quality: Assessments and relationships, Acta Acust. United Ac. **87**, 367–377 (2001)
5.29 L. Gros, N. Chateu, S. Busson: Effects of context on the subjective assessment of time-varying speech quality: listening/conversation, laboratory/real environment, Acta Acust. United Ac. **90**, 1037–1051 (2004)
5.30 L. Gros: The impact of listening and conversational situations on speech perceived quality for time-varying impairments. P: Int. Conf. Measurement of Speech and Audio Quality in Networks, 17–19 (2002)
5.31 J. Rosenbluth: Testing the quality of connections having time varying impairments, Comitee T1 Standards Contribution ANSI T1A1.7/98-031, (1998)
5.32 P. Gray, R. Massara, M. Hollier: An experimental investigation of the accumulation of perceived error in time-varying speech distortions. In Preprint: Audio Engineering Society 103rd Convention (1997)
5.33 ITU-T Rec. P.880: *Continuous evaluation of time varying speech quality* (Geneva 2004)
5.34 S. Quackenbush, T. Barnwell, M. Clements: *Objective Measures of Speech Quality* (Prentice Hall, Englewood Cliffs 1988)
5.35 S. Jayant, P. Noll: *Digital Coding of Waveforms* (Prentice Hall, Englewood Cliffs 1984)
5.36 M. Schroeder, B. Atal, J. Hall: *Objective Measure of Certain Speech Signal Degradations Based on Masking Properties of Human Auditory Perception* (Academic, New York 1979)
5.37 N. Kitawaki, H. Nagabuchi, K. Itoh: Objective quality evaluation for low-bit-rate speech coding systems, IEEE J. Sel. Area. Commun. **6**, 242–248 (1988)
5.38 W.B. Kleijn, K.K. Paliwal (Eds.): *Speech Coding and Synthesis* (Elsevier Science, Amsterdam 1995)
5.39 R. Viswanathan, J. Makhoul, W. Russel: Towards perceptually consistent measures of spectral distance, Proc. IEEE ICASSP **1**, 485–488 (1976)
5.40 S. Dimolitsas: Objective speech distortion measures and their relevance to speech quality measurements, IEE Proc. **136**, 317–324 (1989)
5.41 ITU-T Rec. P.56: *Objective measurement of active speech level* (Geneva 1993)
5.42 E. Zwicker, H. Fastl: *Psycho-Acoustics: Facts and Models* (Springer, New York 1999)
5.43 B.C.J. Moore: *An Introduction to the Psychology of Hearing* (Academic, London 1989)

5.44 T. Dau, D. Püschel, A. Kohlrausch: A quantitative model of the effective signal processing in the auditory system. I. Model structure, J. Acoust. Soc. Am. **99**, 3615–3622 (1996)

5.45 A. Rix, A. Bourret, M. Hollier: Models of human perception, BT Technol. J. **17**, 24–34 (1999)

5.46 S. Voran: A simplified version of the ITU algorithm for objective measurement of speech codec quality, Proc. IEEE ICASSP **1**, 537–540 (1998)

5.47 K. Brandenburg: Evaluation of quality for audio encoding at low bit rates. In preprint: Audio Engineering Society 82nd Convention (1987)

5.48 J. Karjalainen: A new auditory model for the evaluation of sound quality of audio systems, Proc. IEEE ICASSP **10**, 608–611 (1985)

5.49 S. Wang, A. Sekey, A. Gersho: An objective measure for predicting subjective quality of speech coders, IEEE J. Sel. Area. Commun. **10**(5), 819–829 (1992)

5.50 J. Lalou: The information index: An objective measure of speech transmission performance, Ann. Telecommun. **45**, 47–65 (1990)

5.51 R. Bouquin, G. Faucon: Using the coherence function for noise reduction, Proc. IEE **139**(3), 276–282 (1992)

5.52 J. Beerends, J. Stemerdink: A perceptual speech-quality measure based on a psychoacoustic sound representation, J. Audio Eng. Soc. **42**(3), 115–123 (1994)

5.53 C. Colomes, C. Schmidmer, T. Thiede, W. Treurniet: Perceptual quality assessment for digital audio: (PEAQ) – the new ITU standard for objective measurement of the perceived audio quality, In Proc.: AES 17th Int. Conf. 337–351 (1999)

5.54 ITU-R. BS.1387-1: *Method for Objective Measurements of Perceived Audio Quality (PEAQ)* (2001)

5.55 ITU-T Rec. P. 862: *Perceptual evaluation of speech quality (PESQ)* (2001).

5.56 ITU-R Rec. P.862.2: *Wideband extension to Recommendation P.862 for the assessment of wideband telephone networks and speech codecs* (2005)

5.57 H. Knagenhjelm, W.B. Kleijn: Spectral dynamics is more important than spectral distortion, P. IEEE ICASSP **1**, 732–735 (1995)

5.58 F. Norden, T. Eriksson: Time evolution in LPC spectrum coding, IEEE T. Speech Audi. P. **12**, 290–301 (2004)

5.59 T. Quatieri, R. Dunn: Speech enhancement based on auditory spectral change, P. IEEE ICASSP **1**, 257–260 (2002)

5.60 M. Hollier, M. Hawksford, D. Guard: Error activity and error entropy as a measure of psychoacoustic significance in the perceptual domain, IEE P-Vis. Image Sign. **141**(3), 203–208 (1994)

5.61 S. Voran: Objective estimation of perceived speech quality – Part I: Development of the measuring normalizing block technique, IEEE T. Speech Audi. P. **7**(4), 371–382 (1999)

5.62 S. Voran: Objective estimation of perceived speech quality – Part II: Evaluation of the measuring normalizing block technique, IEEE T. Speech Audi. Proc. **7**(4), 383–390 (1999)

5.63 H. Coetzee, T.B. III: An LSP based speech quality measure, Proc. IEEE ICASSP **1**, 596–599 (1989)

5.64 D. Klatt: Prediction of perceived phonetic distance from critical-band spectra: a first step, Proc. IEEE ICASSP **7**, 1278–1281 (1982)

5.65 U. Halka, U. Heute: A new approach to objective quality-measures based on attribute-matching, Speech Commun. **11**, 15–30 (1992)

5.66 W. Zha, W.-Y. Chan: Objective speech quality measurement using statistical data minimg, J. Appl. Signal Process. **9**, 1410–1424 (2005)

5.67 P. Gray, M. Hollier, R. Massara: Non-intrusive speech-quality assessment using vocal-tract models, IEE P-Vis. Image Sign. **147**(6), 493–501 (2000)

5.68 J. Liang, R. Kubichek: Output-based objective speech quality, IEEE 44th Vehicular Technology Conf. **3**(8-10), 1719–1723 (1994)

5.69 H. Hermansky: Perceptual linear prediction (PLP) analysis of speech, J. Acoust. Soc. Am. **87**, 1738–1752 (1990)

5.70 A. Conway: A passive method for monitoring voice-over-IP call quality with ITU-T objective speech quality measurement methods, Proc. IEEE Int. Conf. Commun. **4**, 2583–2586 (2002)

5.71 A. Conway: Output-based method of applying PESQ to measure the perceptual quality of framed speech signals, Proc. IEEE Wireless Commun. Netw. **4**, 2521–2526 (2004)

5.72 D. Kim: ANIQUE: An auditory model for single-ended speech quality estimation, IEEE T. Speech Audi. P. **13**, 821–831 (2005)

5.73 D. Kim, A. Tarraf: Enhanced perceptual model for non-intrusive speech quality assessment, P. IEEE ICASSP **1**, 829–832 (2006)

5.74 O. Au, K. Lam: A novel output-based objective speech quality measure for wireless communication, Signal Process. P, 4th Int. Conf. **1**, 666–669 (1998)

5.75 ITU-T Rec. P.563: *Single ended method for objective speech quality assessment in narrow-band telephony applications* (2004)

5.76 T. Falk, Q. Xu, W.-Y. Chan: Non-intrusive GMM-based speech quality measurement, P. IEEE ICASSP **1**, 125–128 (2005)

5.77 G. Chen, V. Parsa: Bayesian model based non-intrusive speech quality evaluation, P. IEEE ICASSP **1**, 385–388 (2005)

5.78 D. Picovici, A. Mahdi: Output-based objective speech quality measure using self-organizing map, P. IEEE ICASSP **1**, 476–479 (2003)

5.79 V. Grancharov, D. Y. Zhao, J. Lindblom, W. B. Kleijn: Low complexity, non-intrusive speech quality assessment, IEEE Trans. Speech Audio. Process. *14*, 1948-1956 (2006)

5.80 ITU-T Rec. G.107: *The e-model, a computational model for use in transmission planning* (Geneva 2005)

5.81 ITU-T Rec. P. Supplement 3: *Models for predicting transmission quality from objective measurements (Withdrawn 1997)* International Telecommunication Union (1993)
5.82 R. Appel, J. Beerends: On the quality of hearing one's own voice, J. Audio Eng. Soc. **50**(4), 237–248 (2002)
5.83 ITU-T Rec. P. Supplement 23: ITU-T coded-speech database, International Telecommunication Union (1998)
5.84 A. Rix, J. Beerends, M. Hollier, A. Hekstra: Perceptual evaluation of speech quality (PESQ) – A new method for speech quality assessment of telephone networks and codecs, P. IEEE ICASSP **2**, 749–752 (2001)
5.85 T. Goldstein, A.W. Rix: Subjective comparison of speech enhancement algorithms, P. IEEE ICASSP **3**, 1064–1067 (2004)
5.86 A.W. Rix: Perceptual speech quality assessment – a review, P. IEEE ICASSP **3**, 1056–1059 (2004)
5.87 M. Werner, T. Junge, P. Vary: Quality control for AMR speech channels in GSM networks, P. IEEE ICASSP **3**, 1076–1079 (2004)
5.88 ITU-T Rec. P.920: Interactive test methods for audiovisual communications, (2000)
5.89 http://www.itu.int/ITU-T/studygroups/com12/index.asp

13. Adaptive Blind Multichannel Identification

Y. Huang, J. Benesty, J. Chen

Blind multichannel identification was first introduced in the mid 1970s and initially studied in the communication society with the intention of designing more-efficient communication systems by avoiding a training phase. Recently this idea has become increasingly interesting for acoustics and speech processing research, driven by the fact that in most acoustic applications for speech processing and communication very little or nothing is known about the source signals. Since human ears have an extremely wide dynamic range and are much more sensitive to weak tails of the acoustic impulse responses, these impulse responses need to be modeled using fairly long filters. Attempting to identify such a multichannel system blindly with a batch method involves intensive computational complexity. This is not just bad system design, but technically rather implausible, particularly for real-time systems. Therefore, adaptive blind multichannel identification algorithms are favorable and pragmatically useful. This chapter describes some fundamental issues in blind multichannel identification and reviews a number of state-of-the-art adaptive algorithms.

13.1 Overview

The need for identifying an acoustic multichannel system arises in a variety of speech processing and communication applications. While in some cases a reference source signal is known (acoustic echo cancelation is a typical example), for the majority of the applications (e.g., speech dereverberation, time delay estimation for source localization, source separation, speech enhancement, etc.) a priori knowledge of the source speech signal is either inaccessible or very expensive to acquire. This makes blind multichannel identification methods necessary and desirable.

The innovative idea of blind system identification was first proposed by *Sato* in [13.1]. Early studies of blind channel identification and equalization focused primarily on methods based on higher (than second)-order statistics (HOS) [13.2–4] (see [13.5] for a tutorial on this class of approaches). Because HOS cannot be accurately computed from a small number of observations, slow convergence is the critical drawback of all existing HOS methods. In addition, a cost function based on the HOS is barely convex and an HOS algorithm can be easily misled to a local minimum by corrupting noise in the observations. After it was recognized that the problem can be solved in the light of only second-order statistics (SOS) of system outputs [13.6], the focus of the blind multichannel identification research has shifted to SOS methods.

Using SOS to blindly identify a linear system requires that the number of outputs would be greater than the number of inputs. Therefore, only single-input multiple-output (SIMO) and multiple-input multiple-output (MIMO) acoustic systems are of interest in this problem. Blind MIMO identification, as of today, is still an open research problem and, in this handbook, only reviews of supposedly well-understood techniques are included. As a result, the focus will be on acoustic SIMO systems.

There is already a large literature on SOS-based blind SIMO identification algorithms with numerous variants. Celebrated work includes the cross-relation (CR) algorithm [13.7–10], the subspace algorithm [13.11], the linear prediction-based subspace algorithm [13.12], and the two-step maximum-likelihood algorithm [13.13], (see [13.14] for a more-comprehensive survey and a more-systematic classification of SOS-based blind SIMO identification algorithms). These *batch* methods can accurately estimate an identifiable acoustic SIMO system using a finite number of samples when additive noise in the system outputs is weak. However, they are in general computationally intensive and it was believed that they would be difficult to implement in an adaptive mode [13.14]. For blind multichannel identification to be practically useful in real-time speech applications, it is imperative to have some algorithms that are computationally simple and can be adaptively implemented. In this chapter, we will present a review of the state of the art of *adaptive* blind SIMO identification algorithms.

This Chapter is organized as follows. Section 13.2 describes the SIMO model used in this review and formulates the problem of blind multichannel identification. Section 13.3 presents the sufficient and necessary conditions for an acoustic SIMO to be blindly identifiable using only SOS and explains the principle of SOS-based blind SIMO identification techniques. In Sect. 13.4, the time-domain unit-norm constrained multichannel LMS (least mean square) (MCLMS) and Newton algorithms are developed. In Sect. 13.5, an optimal step size is deduced for the unconstrained MCLMS algorithm, which leads to the variable-step-size unconstrained MCLMS algorithm. Section 13.6 discusses the frequency-domain blind multichannel identification algorithms. Section 13.7 explores how to take advantage of spareness in acoustic impulse responses for faster convergence in blind multichannel identification methods. Finally, we provide a brief summary in Sect. 13.8.

13.2 Signal Model and Problem Formulation

This chapter considers an acoustic SIMO system in which there is one speech source and N microphones, as illustrated in Fig. 13.1. The n-th microphone output is expressed as:

$$x_n(k) = h_n * s(k) + b_n(k), \quad n = 1, 2, \cdots, N, \quad (13.1)$$

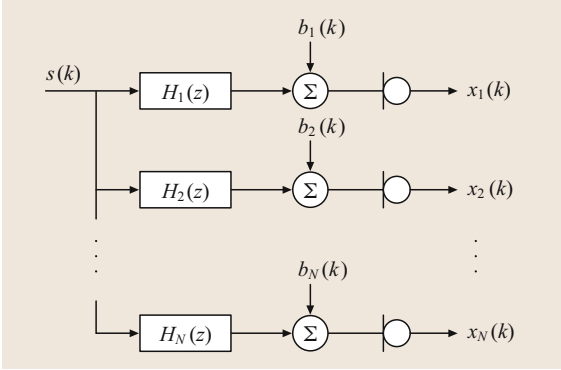


Fig. 13.1 Illustration of an acoustic SIMO system

where h_n is the channel impulse response from the source to the n-th microphone, the symbol $*$ denotes the linear convolution operator, $s(k)$ is the source signal at time k, and $b_n(k)$ is the additive noise at the n-th microphone output. The channel impulse responses are delineated with finite impulse response (FIR) filters. The additive noise components in different channels are assumed to be uncorrelated with the source signal even though they might be mutually dependent.

In vector/matrix form, the SIMO signal model (13.1) is written as:

$$\boldsymbol{x}_n(k) = \boldsymbol{H}_n \cdot \boldsymbol{s}(k) + \boldsymbol{b}_n(k), \quad n = 1, 2, \cdots, N, \quad (13.2)$$

where

$$\boldsymbol{x}_n(k) = [x_n(k) \;\; x_n(k-1) \;\; \cdots \;\; x_n(k-L+1)]^{\mathrm{T}},$$

$$\boldsymbol{H}_n = \begin{pmatrix} h_{n,0} & \cdots & h_{n,L-1} & 0 & \cdots & 0 \\ 0 & h_{n,0} & \cdots & h_{n,L-1} & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h_{n,0} & \cdots & h_{n,L-1} \end{pmatrix}_{L\times(2L-1)},$$

$$\boldsymbol{s}(k) = [s(k)\ \ s(k-1) \\ \cdots\ s(k-L+1)\ \cdots\ s(k-2L+2)]^{\mathrm{T}},$$
$$\boldsymbol{b}_n(k) = [b_n(k)\ \ b_n(k-1)\ \cdots\ b_n(k-L+1)]^{\mathrm{T}},$$

$[\cdot]^{\mathrm{T}}$ denotes the transpose of a matrix or a vector, and L is the length of the longest channel impulse response in such a SIMO system.

Therefore, mathematically speaking, the blind multichannel identification problem is to estimate the channel impulse responses

$$\boldsymbol{h}_n = [h_{n,0}\ \ h_{n,1}\ \cdots\ h_{n,L-1}]^{\mathrm{T}}, \quad n = 1, 2, \cdots, N,$$

from the observation $x_n(k)$ without knowledge of the source signal $\boldsymbol{s}(k)$.

13.3 Identifiability and Principle

An acoustic SIMO system is not always blindly identifiable using only the second-order statistics (SOS) of the microphone outputs. A multichannel FIR system can be blindly identified primarily because of the channel diversity. As an extreme counterexample, if all channels of a SIMO system are identical, the system reduces to a single-input single-output (SISO) system, which one can easily tell is blindly unidentifiable using only SOS. In addition, the source signal needs to have sufficient modes to make the channels fully excited. According to [13.10], two inductive conditions (one on the channel diversity and the other on the input signals) are necessary and sufficient to ensure blind SIMO identifiability, and are shared by all SOS-based blind SIMO identification methods:

1. The polynomials formed from $\boldsymbol{h}_n$ $(n = 1, 2, \cdots, N)$ are co-prime, i. e., the corresponding channel transfer functions $H_n(z) = \sum_{l=0}^{L-1} h_{n,l} z^{-l}$ do not share any common zeros;
2. The autocorrelation matrix of the input signal $\boldsymbol{R}_{ss} = E\{\boldsymbol{s}(k)\boldsymbol{s}^{\mathrm{T}}(k)\}$, where $E\{\cdot\}$ denotes mathematical expectation, is of full rank (such that the SIMO system can be fully excited).

In the rest of this chapter, these two conditions are assumed to hold so that we will be dealing with a blindly identifiable FIR SIMO system.

The easiest way to show how a SIMO system can be blindly identified probably is to use the cross-relation method. From (13.1), we derive the fact that, in the absence of noise,

$$x_i * h_j = s * h_i * h_j = x_j * h_i, \\ i, j = 1, 2, \cdots, N, \quad i \neq j\,. \tag{13.3}$$

At time k, we then have

$$\boldsymbol{x}_i^{\mathrm{T}}(k)\boldsymbol{h}_j = \boldsymbol{x}_j^{\mathrm{T}}(k)\boldsymbol{h}_i\,, \quad i, j = 1, 2, \cdots, N, \quad i \neq j\,. \tag{13.4}$$

Multiplying (13.4) by $\boldsymbol{x}_i(k)$ from the left side and taking expectation yields,

$$\boldsymbol{R}_{x_i x_i}\boldsymbol{h}_j = \boldsymbol{R}_{x_i x_j}\boldsymbol{h}_i\,, \quad i, j = 1, 2, \cdots, N, \quad i \neq j\,, \tag{13.5}$$

where

$$\boldsymbol{R}_{x_i x_j} = E\left\{\boldsymbol{x}_i(k)\boldsymbol{x}_j^{\mathrm{T}}(k)\right\}.$$

Formula (13.5) consists of $N(N-1)$ distinct equations. By summing up the $N-1$ cross-relations associated with one particular channel h_j, we obtain

$$\sum_{i=1, i\neq j}^{N} \boldsymbol{R}_{x_i x_i}\boldsymbol{h}_j = \sum_{i=1, i\neq j}^{N} \boldsymbol{R}_{x_i x_j}\boldsymbol{h}_i\,, \\ j = 1, 2, \cdots, N\,. \tag{13.6}$$

Over all channels, we then have a total of N equations. In matrix form, this set of equations is written:

$$\boldsymbol{R}_{x+}\boldsymbol{h} = \boldsymbol{0}\,, \tag{13.7}$$

where

$$\boldsymbol{R}_{x+} = \begin{pmatrix} \sum_{n\neq 1}\boldsymbol{R}_{x_n x_n} & -\boldsymbol{R}_{x_2 x_1} & \cdots & -\boldsymbol{R}_{x_N x_1} \\ -\boldsymbol{R}_{x_1 x_2} & \sum_{n\neq 2}\boldsymbol{R}_{x_n x_n} & \cdots & -\boldsymbol{R}_{x_N x_2} \\ \vdots & \vdots & \ddots & \vdots \\ -\boldsymbol{R}_{x_1 x_N} & -\boldsymbol{R}_{x_2 x_N} & \cdots & \sum_{n\neq N}\boldsymbol{R}_{x_n x_n} \end{pmatrix},$$

$$\boldsymbol{h} = \left[\boldsymbol{h}_1^{\mathrm{T}}\ \ \boldsymbol{h}_2^{\mathrm{T}}\ \cdots\ \boldsymbol{h}_N^{\mathrm{T}}\right]^{\mathrm{T}}.$$

If the SIMO system is blindly identifiable, the matrix $\boldsymbol{R}_{x+}$ is rank deficient by 1 (in the absence of noise) and the channel impulse responses $\boldsymbol{h}$ can be uniquely determined up to a scale.

When additive noise is present, the right-hand side of (13.7) is no longer zero and an error vector is produced:

$$\boldsymbol{e} = \boldsymbol{R}_{x+}\boldsymbol{h}\,. \tag{13.8}$$

This error can be used to define a cost function

$$J = \|\boldsymbol{e}\|^2 = \boldsymbol{e}^{\mathrm{T}}\boldsymbol{e}\,. \tag{13.9}$$

We can then determine a vector $\hat{\boldsymbol{h}}$ as the solution by minimizing this cost function in the least-squares sense:

$$\hat{\boldsymbol{h}} = \arg\min_{\boldsymbol{h}} J = \arg\min_{\boldsymbol{h}} \boldsymbol{h}^{\mathrm{T}} \boldsymbol{R}_{x+}^{\mathrm{T}} \boldsymbol{R}_{x+} \boldsymbol{h} \,. \tag{13.10}$$

In this case, $\boldsymbol{R}_{x+}$ is positive definite rather than positive semidefinite and the desired estimate $\hat{\boldsymbol{h}}$ would be the eigenvector of $\boldsymbol{R}_{x+}$ corresponding to its smallest eigenvalue (here we assume that the noise is white, incoherent or uncorrelated, and weaker than the source signal).

Note that the estimated channel impulse response vector is aligned to the true one, but up to a nonzero scale. This inherent scale ambiguity is usually harmless in most, if not all, of acoustic signal processing applications. But in the development of an adaptive algorithm, attention needs to be paid to prevent convergence to a trivial all-zero estimate. This will become clearer after the reader finishes reading this chapter.

13.4 Constrained Time-Domain Multichannel LMS and Newton Algorithms

In this section, we intend to develop two time-domain *adaptive* algorithms for blind SIMO identification, namely the multichannel LMS (MCLMS) and multichannel Newton (MCN) algorithms [13.15].

13.4.1 Unit-Norm Constrained Multichannel LMS Algorithm

To begin, we use the cross-relations between the i-th and j-th outputs given by (13.4). When noise is present and/or the estimate of channel impulse responses deviates from the true value, an a priori error signal is produced:

$$e_{ij}(k+1) = \boldsymbol{x}_i^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_j(k) - \boldsymbol{x}_j^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_i(k) \,, \quad i, j = 1, 2, \cdots, N \,, \tag{13.11}$$

where $\hat{\boldsymbol{h}}_i(k)$ is the model filter for the i-th channel at time k. Assuming that these error signals are equally important, we now define a cost function as follows:

$$\chi(k+1) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} e_{ij}^2(k+1) \,, \tag{13.12}$$

where we exclude the cases of $e_{ii}(k) = 0$ $(i = 1, 2, \cdots, N)$ and count the $e_{ij}(k) = -e_{ij}(k)$ pair only once.

In order to avoid a trivial estimate with all zero elements, a constraint should be imposed on the model filters $\hat{\boldsymbol{h}}(k)$. Two constraints were proposed. One is the easily understood unit-norm constraint, i. e., $\|\hat{\boldsymbol{h}}(k)\| = 1$. The other is the component-normalization constraint, i. e., $\boldsymbol{c}^{\mathrm{T}}\hat{\boldsymbol{h}}(k) = 1$, where $\boldsymbol{c}$ is a constant vector. As an example, if we know that one coefficient, say $h_{n,l}$ $(n = 1, 2, \cdots, N,\ l = 0, 1, \cdots, L-1)$, is equal to α, which is not zero, then we may properly specify $\boldsymbol{c} = [0, \cdots, 1/\alpha, \cdots, 0]^{\mathrm{T}}$ with $1/\alpha$ being the $(nL+l)$-th element of $\boldsymbol{c}$. Even though the component-normalization constraint can be more robust to noise than the unit-norm constraint [13.16], knowledge of the location of the constrained component $h_{n,l}$ and its value α may not be available in practice. So the unit-norm constraint was more widely used and will be employed in this chapter.

With the unit-norm constraint enforced on $\hat{\boldsymbol{h}}(k)$, the normalized error signal is obtained:

$$\epsilon_{ij}(k+1) = e_{ij}(k+1)/\|\hat{\boldsymbol{h}}(k)\| \,. \tag{13.13}$$

Accordingly, the cost function is formulated as:

$$J(k+1) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \epsilon_{ij}^2(k+1) = \frac{\chi(k+1)}{\|\hat{\boldsymbol{h}}(k)\|^2} \,. \tag{13.14}$$

The update equation of the unit-norm-constrained MCLMS algorithm is then given by

$$\hat{\boldsymbol{h}}(k+1) = \hat{\boldsymbol{h}}(k) - \mu \nabla J(k+1) \,, \tag{13.15}$$

where μ is a small positive step size and $\nabla J(k+1)$ denotes the gradient of $J(k+1)$ with respect to $\hat{\boldsymbol{h}}(k)$. The gradient is determined as follows:

$$\begin{aligned} \nabla J(k+1) &= \frac{\partial J(k+1)}{\partial \hat{\boldsymbol{h}}(k)} = \frac{\partial}{\partial \hat{\boldsymbol{h}}(k)} \left(\frac{\chi(k+1)}{\|\hat{\boldsymbol{h}}(k)\|^2} \right) \\ &= \frac{\partial}{\partial \hat{\boldsymbol{h}}(k)} \left(\frac{\chi(k+1)}{\hat{\boldsymbol{h}}^{\mathrm{T}}(k)\hat{\boldsymbol{h}}(k)} \right) \\ &= \frac{1}{\|\hat{\boldsymbol{h}}(k)\|^2} \\ &\quad \times \left(\frac{\partial \chi(k+1)}{\partial \hat{\boldsymbol{h}}(k)} - 2J(k+1)\hat{\boldsymbol{h}}(k) \right) \,, \end{aligned} \tag{13.16}$$

where

$$\frac{\partial \chi(k+1)}{\partial \hat{\boldsymbol{h}}(k)} = \left[\left(\frac{\partial \chi(k+1)}{\partial \hat{\boldsymbol{h}}_1(k)} \right)^{\mathrm{T}} \left(\frac{\partial \chi(k+1)}{\partial \hat{\boldsymbol{h}}_2(k)} \right)^{\mathrm{T}} \cdots \left(\frac{\partial \chi(k+1)}{\partial \hat{\boldsymbol{h}}_N(k)} \right)^{\mathrm{T}} \right]^{\mathrm{T}} \,.$$

Let us now evaluate the partial derivative of $\chi(k+1)$ with respect to the coefficients of the n-th ($n = 1, 2, \cdots, N$) channel impulse response:

$$\begin{aligned}\frac{\partial \chi(k+1)}{\partial \hat{\boldsymbol{h}}_n(k)} &= \frac{\partial\left[\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} e_{ij}^2(k+1)\right]}{\partial \hat{\boldsymbol{h}}_n(k)} \\ &= \sum_{i=1}^{n-1} 2e_{in}(k+1)\boldsymbol{x}_i(k+1) \\ &\quad + \sum_{j=n+1}^{N} 2e_{nj}(k+1)[-\boldsymbol{x}_j(k+1)] \\ &= \sum_{i=1}^{n-1} 2e_{in}(k+1)\boldsymbol{x}_i(k+1) \\ &\quad + \sum_{j=n+1}^{N} 2e_{jn}(k+1)\boldsymbol{x}_j(k+1) \\ &= \sum_{i=1}^{N} 2e_{in}(k+1)\boldsymbol{x}_i(k+1)\,, \end{aligned} \tag{13.17}$$

where the last step follows from the fact that $e_{nn}(k+1) = 0$. We may express this equation concisely in matrix form as

$$\begin{aligned}\frac{\partial \chi(k+1)}{\partial \hat{\boldsymbol{h}}_n(k)} &= 2\boldsymbol{X}(k+1)\boldsymbol{e}_n(k+1) \\ &= 2\boldsymbol{X}(k+1)[\boldsymbol{C}_n(k+1) \\ &\quad - \boldsymbol{D}_n(k+1)]\hat{\boldsymbol{h}}(k)\,, \end{aligned} \tag{13.18}$$

where we have defined, for convenience,

$$\begin{aligned}\boldsymbol{X}(k+1) &= (\boldsymbol{x}_1(k+1)\ \boldsymbol{x}_2(k+1) \\ &\quad \cdots\ \boldsymbol{x}_N(k+1))_{L\times N}\,, \\ \boldsymbol{e}_n(k+1) &= [e_{1n}(k+1)\ e_{2n}(k+1) \\ &\quad \cdots e_{Nn}(k+1)]^{\mathrm{T}} \\ &= \begin{pmatrix} \boldsymbol{x}_1^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_n(k) - \boldsymbol{x}_n^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_1(k) \\ \boldsymbol{x}_2^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_n(k) - \boldsymbol{x}_n^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_2(k) \\ \vdots \\ \boldsymbol{x}_N^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_n(k) - \boldsymbol{x}_n^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_N(k) \end{pmatrix} \\ &= [\boldsymbol{C}_n(k+1) - \boldsymbol{D}_n(k+1)]\hat{\boldsymbol{h}}(k)\,, \end{aligned}$$

and

$$\begin{aligned}\boldsymbol{C}_n(k+1) &= \begin{pmatrix} \boldsymbol{0}\ \cdots\ \boldsymbol{0}\ \boldsymbol{x}_1^{\mathrm{T}}(k+1)\ \boldsymbol{0}\ \cdots\ \boldsymbol{0} \\ \boldsymbol{0}\ \cdots\ \boldsymbol{0}\ \boldsymbol{x}_2^{\mathrm{T}}(k+1)\ \boldsymbol{0}\ \cdots\ \boldsymbol{0} \\ \vdots\ \cdots\ \vdots\ \ \vdots\ \ \vdots\ \cdots\ \vdots \\ \underbrace{\boldsymbol{0}\ \cdots\ \boldsymbol{0}}_{(n-1)L}\ \boldsymbol{x}_N^{\mathrm{T}}(k+1)\ \underbrace{\boldsymbol{0}\ \cdots\ \boldsymbol{0}}_{(N-n)L} \end{pmatrix}_{N\times NL} \\ &= \left(\boldsymbol{0}_{N\times(n-1)L}\ \ \boldsymbol{X}^{\mathrm{T}}(k+1)\ \ \boldsymbol{0}_{N\times(N-n)L}\right), \\ \boldsymbol{D}_n(k+1) &= \begin{pmatrix} \boldsymbol{x}_n^{\mathrm{T}}(k+1) & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{x}_n^{\mathrm{T}}(k+1) & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{x}_n^{\mathrm{T}}(k+1) \end{pmatrix}_{N\times NL}. \end{aligned}$$

Continuing, we evaluate the two matrix products in (13.18) individually as follows:

$$\begin{aligned}&\boldsymbol{X}(k+1)\boldsymbol{C}_n(k+1) \\ &\quad = \boldsymbol{X}(k+1) \\ &\qquad \cdot \left(\boldsymbol{0}_{N\times(n-1)L}\ \ \boldsymbol{X}^{\mathrm{T}}(k+1)\ \ \boldsymbol{0}_{N\times(N-n)L}\right)_{N\times NL} \\ &\quad = \Bigg(\boldsymbol{0}_{L\times(n-1)L}\ \ \sum_{i=1}^{N}\tilde{\boldsymbol{R}}_{x_i x_i}(k+1) \\ &\qquad \boldsymbol{0}_{L\times(N-n)L}\Bigg)_{L\times NL}\,, \end{aligned} \tag{13.19}$$

$$\begin{aligned}&\boldsymbol{X}(k+1)\boldsymbol{D}_n(k+1) \\ &\quad = (\tilde{\boldsymbol{R}}_{x_1 x_n}(k+1)\ \ \tilde{\boldsymbol{R}}_{x_2 x_n}(k+1) \\ &\qquad \cdots\ \tilde{\boldsymbol{R}}_{x_N x_n}(k+1))_{L\times NL}\,, \end{aligned} \tag{13.20}$$

where

$$\begin{aligned}\tilde{\boldsymbol{R}}_{x_i x_j}(k+1) &= \boldsymbol{x}_i(k+1)\boldsymbol{x}_j^{\mathrm{T}}(k+1)\,, \\ & i, j = 1, 2, \cdots, N. \end{aligned}$$

Here we put a tilde on $\tilde{\boldsymbol{R}}_{x_i x_j}$ to distinguish this instantaneous value from its mathematical expectation $\boldsymbol{R}_{x_i x_j}$.

Next, substituting (13.19) and (13.20) into (13.18) yields

$$\begin{aligned}\frac{\partial \chi(k+1)}{\partial \hat{\boldsymbol{h}}_n(k)} &= 2\Big(-\tilde{\boldsymbol{R}}_{x_1 x_n}(k+1) \\ &\qquad \cdots \sum_{i\neq n}\tilde{\boldsymbol{R}}_{x_i x_i}(k+1) \\ &\qquad \cdots -\tilde{\boldsymbol{R}}_{x_N x_n}(k+1)\Big)\hat{\boldsymbol{h}}(k)\,. \end{aligned} \tag{13.21}$$

Therefore, we incorporate (13.21) into (13.16) and obtain

$$\frac{\partial \chi(k+1)}{\partial \hat{\boldsymbol{h}}(k)} = 2\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k)\,, \tag{13.22}$$

$$\nabla J(k+1) = \frac{1}{\|\hat{\boldsymbol{h}}(k)\|^2} \cdot [2\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - 2J(k+1)\hat{\boldsymbol{h}}(k)]\,, \tag{13.23}$$

where

$$\tilde{\boldsymbol{R}}_{x+}(k) = \begin{pmatrix} \sum_{n\neq 1}\tilde{\boldsymbol{R}}_{x_n x_n}(k) & -\tilde{\boldsymbol{R}}_{x_2 x_1}(k) & \cdots & -\tilde{\boldsymbol{R}}_{x_N x_1}(k) \\ -\tilde{\boldsymbol{R}}_{x_1 x_2}(k) & \sum_{n\neq 2}\tilde{\boldsymbol{R}}_{x_n x_n}(k) & \cdots & -\tilde{\boldsymbol{R}}_{x_N x_2}(k) \\ \vdots & \vdots & \ddots & \vdots \\ -\tilde{\boldsymbol{R}}_{x_1 x_N}(k) & -\tilde{\boldsymbol{R}}_{x_2 x_N}(k) & \cdots & \sum_{n\neq N}\tilde{\boldsymbol{R}}_{x_n x_n}(k) \end{pmatrix}.$$

Finally, we substitute (13.23) into (13.15) and deduce the update equation of the unit-norm-constrained MCLMS algorithm:

$$\hat{\boldsymbol{h}}(k+1) = \hat{\boldsymbol{h}}(k) - \frac{2\mu}{\|\hat{\boldsymbol{h}}(k)\|^2} [\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - J(k+1)\hat{\boldsymbol{h}}(k)]\,. \tag{13.24}$$

If the channel estimate is always normalized after each update, then we have the simplified, unit-norm-constrained MCLMS algorithm [13.15]:

$$\hat{\boldsymbol{h}}(k+1) = \{\hat{\boldsymbol{h}}(k) - 2\mu[\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - \chi(k+1)\hat{\boldsymbol{h}}(k)]\} \Big/ \{\|\hat{\boldsymbol{h}}(k) - 2\mu[\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - \chi(k+1)\hat{\boldsymbol{h}}(k)]\|\}\,. \tag{13.25}$$

The time-domain, unit-norm-constrained MCLMS adaptive algorithm is summarized in Table 13.1.

Now we would like to comment briefly on the convergence of the unit-norm-constrained MCLMS algorithm. Assuming that the independence assumption [13.17] holds, it can be easily shown that the unit-norm-constrained MCLMS algorithm converges in the mean if the step size satisfies

$$0 < \mu < \frac{1}{\lambda_{\max}}\,, \tag{13.26}$$

where $\lambda_{\max}$ is the largest eigenvalue of the matrix $E\{\tilde{\boldsymbol{R}}_{x+}(k) - J(k)\boldsymbol{I}_{NL\times NL}\}$ and $\boldsymbol{I}_{NL\times NL}$ is the identity matrix of size NL by NL. After convergence, taking the expectation of (13.24) produces

$$\boldsymbol{R}_{x+}\frac{\hat{\boldsymbol{h}}(\infty)}{\|\hat{\boldsymbol{h}}(\infty)\|} = E\{J(\infty)\}\frac{\hat{\boldsymbol{h}}(\infty)}{\|\hat{\boldsymbol{h}}(\infty)\|}\,, \tag{13.27}$$

which is the desired result: $\hat{\boldsymbol{h}}$ converges in the mean to the eigenvector of $\boldsymbol{R}_{x+}$ corresponding to the smallest eigenvalue $E\{J(\infty)\}$.

Table 13.1 The unit-norm-constrained multichannel LMS adaptive algorithm for the blind identification of an SIMO FIR system

Parameters:	$\hat{\boldsymbol{h}} = [\hat{\boldsymbol{h}}_1^{\mathrm{T}}\ \hat{\boldsymbol{h}}_2^{\mathrm{T}}\ \cdots\ \hat{\boldsymbol{h}}_N^{\mathrm{T}}]^{\mathrm{T}}$, model filter
	$\mu > 0$, step size
Initialization:	$\hat{\boldsymbol{h}}_n(0) = [1\ \ 0\ \ \cdots\ \ 0]^{\mathrm{T}}$, $n = 1, 2, \cdots, N$
	$\hat{\boldsymbol{h}}(0) = \hat{\boldsymbol{h}}(0)/\sqrt{N}$ (normalization)
Computation:	For $k = 0, 1, 2, \cdots$, compute
	(a) $e_{ij}(k+1) = \boldsymbol{x}_i^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_j(k) - \boldsymbol{x}_j^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_i(k)$ $i, j = 1, 2, \cdots, N$
	(b) $\chi(k+1) = \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} e_{ij}^2(k+1)$
	(c) Construct the matrix $\tilde{\mathbf{R}}_{x+}(k+1)$ following Sect. 13.4.2
	(d) $\hat{\boldsymbol{h}}(k+1) = \dfrac{\hat{\boldsymbol{h}}(k) - 2\mu[\tilde{\mathbf{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - \chi(k+1)\hat{\boldsymbol{h}}(k)]}{\left\|\hat{\boldsymbol{h}}(k) - 2\mu[\tilde{\mathbf{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - \chi(k+1)\hat{\boldsymbol{h}}(k)]\right\|}$

13.4.2 Unit-Norm Constrained Multichannel Newton Algorithm

It has been proved that the unit-norm-constrained MCLMS algorithm developed above can converge in the mean to the desired channel impulse responses. However, one of the difficulties in the design and implementation of the adaptive multichannel LMS filters is the selection of the step size μ. To select the step size μ in an LMS algorithm, there is a trade-off, as pointed out in many studies, between the rate of convergence, the amount of excess mean-square error, and the ability of the algorithm to track the system as its impulse responses change. In order to achieve a good balance of the three competing design objectives, we present here the unit-norm-constrained multichannel Newton algorithm (see [13.18] for the Newton method) with a variable step size during adaptation:

$$\hat{\boldsymbol{h}}(k+1) = \hat{\boldsymbol{h}}(k) - E^{-1}\{\nabla^2 J(k+1)\}\nabla J(k+1)\,, \tag{13.28}$$

where $\nabla^2 J(k+1)$ is the Hessian matrix of $J(k+1)$ with respect to $\hat{\boldsymbol{h}}(k)$. Taking derivative of (13.23) with respect to $\hat{\boldsymbol{h}}(k)$ and using the formula

$$\begin{aligned}
&\frac{\partial}{\partial \hat{\boldsymbol{h}}(k)}[J(k+1)\hat{\boldsymbol{h}}(k)] \\
&= \hat{\boldsymbol{h}}(k)\left[\frac{\partial J(k+1)}{\partial \hat{\boldsymbol{h}}(k)}\right]^{\mathrm{T}} + J(k+1)\boldsymbol{I}_{NL\times NL} \\
&= \hat{\boldsymbol{h}}(k)[\nabla J(k+1)]^{\mathrm{T}} + J(k+1)\boldsymbol{I}_{NL\times NL}\,,
\end{aligned} \tag{13.29}$$

we obtain

$$\begin{aligned}
&\nabla^2 J(k+1) \\
&= \frac{2\tilde{\boldsymbol{R}}_{x+}(k+1)}{\|\hat{\boldsymbol{h}}(k)\|^2} \\
&\quad - \frac{2\left[\hat{\boldsymbol{h}}(k)\nabla J(k+1))^{\mathrm{T}} + J(k+1)\boldsymbol{I}_{NL\times NL}\right]}{\|\hat{\boldsymbol{h}}(k)\|^2} \\
&\quad - \frac{4[\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - J(k+1)\hat{\boldsymbol{h}}(k)]\hat{\boldsymbol{h}}^{\mathrm{T}}(k)}{\|\hat{\boldsymbol{h}}(k)\|^4}\,.
\end{aligned} \tag{13.30}$$

Table 13.2 The unit-norm-constrained multichannel Newton adaptive algorithm for the blind identification of an SIMO FIR system

Parameters:	$\hat{\boldsymbol{h}} = [\hat{\boldsymbol{h}}_1^{\mathrm{T}}\ \hat{\boldsymbol{h}}_2^{\mathrm{T}}\ \cdots\ \hat{\boldsymbol{h}}_N^{\mathrm{T}}]^{\mathrm{T}}$, model filter
	$0 < \rho < 1$, step size
	$0 < \lambda < 1$, forgetting factor
Initialization:	$\hat{\boldsymbol{h}}_n(0) = [1\ \ 0\ \ \cdots\ \ 0]^{\mathrm{T}}$, $n = 1, 2, \cdots, N$
	$\hat{\boldsymbol{h}}(0) = \hat{\boldsymbol{h}}(0)/\sqrt{N}$ (normalization)
Computation:	For $k = 0, 1, 2, \cdots$, compute
	(a) $e_{ij}(k+1) = \boldsymbol{x}_i^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_j(k) - \boldsymbol{x}_j^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_i(k)$, $i, j = 1, 2, \cdots, N$
	(b) $\chi(k+1) = \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} e_{ij}^2(k+1)$
	(c) Construct the matrix $\tilde{\mathbf{R}}_{x+}(k+1)$ following Sect. 13.4.2
	(d) $\hat{\mathbf{R}}_{x+}(k+1) = \begin{cases} \mathrm{diag}\{\sigma_{x_1}^2, \cdots, \sigma_{x_1}^2, \cdots, \sigma_{x_N}^2, \cdots, \sigma_{x_N}^2\}, & k = 0 \\ \lambda\hat{\mathbf{R}}_{x+}(k) + \tilde{\mathbf{R}}_{x+}(k+1), & k \geq 1 \end{cases}$
	(e) $\mathbf{V}(k+1) = 2\hat{\mathbf{R}}_{x+}(k+1) - 4\hat{\boldsymbol{h}}(k)\hat{\boldsymbol{h}}(k)^{\mathrm{T}}\hat{\mathbf{R}}_{x+}(k+1) - 4\hat{\mathbf{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k)\hat{\boldsymbol{h}}(k)^{\mathrm{T}}$
	(f) $\hat{\boldsymbol{h}}(k+1) = \dfrac{\hat{\boldsymbol{h}}(k) - 2\rho\mathbf{V}^{-1}(k+1)[\tilde{\mathbf{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - \chi(k+1)\hat{\boldsymbol{h}}(k)]}{\left\|\hat{\boldsymbol{h}}(k) - 2\rho\mathbf{V}^{-1}(k+1)[\tilde{\mathbf{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - \chi(k+1)\hat{\boldsymbol{h}}(k)]\right\|}$

With the unit-norm constraint $\|\hat{\boldsymbol{h}}(k)\| = 1$, (13.30) can be simplified as follows:

$$\begin{aligned}\nabla^2 J(k+1) = 2\{&\tilde{\boldsymbol{R}}_{x+}(k+1) - \hat{\boldsymbol{h}}(k)[\nabla J(k+1)]^{\mathrm{T}} \\ &- J(k+1)\boldsymbol{I}_{NL\times NL}\} \\ &- 4[\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) \\ &- J(k+1)\hat{\boldsymbol{h}}(k)]\hat{\boldsymbol{h}}^{\mathrm{T}}(k)\,. \end{aligned} \tag{13.31}$$

Taking the mathematical expectation of (13.31) and invoking the independence assumption [13.17] produces

$$\begin{aligned}E\{\nabla^2 J(k+1)\} = &2\boldsymbol{R}_{x+} - 4\hat{\boldsymbol{h}}(k)\hat{\boldsymbol{h}}^{\mathrm{T}}(k)\boldsymbol{R}_{x+} \\ &- 4\boldsymbol{R}_{x+}\hat{\boldsymbol{h}}(k)\hat{\boldsymbol{h}}^{\mathrm{T}}(k) \\ &- 2E\{J(k+1)\} \\ &\cdot[\boldsymbol{I}_{NL\times NL} - 4\hat{\boldsymbol{h}}(k)\hat{\boldsymbol{h}}^{\mathrm{T}}(k)]\,. \end{aligned} \tag{13.32}$$

In practice, $\boldsymbol{R}_{x+}$ and $E\{J(k+1)\}$ are not known such that we have to estimate their values. Since $J(k+1)$ decreases as adaptation proceeds and is relatively small, particularly after convergence, we can neglect the term $E\{J(k+1)\}$ in (13.32) for simplicity and with appropriate accuracy, as suggested by simulations. The matrix $\boldsymbol{R}_{x+}$ is estimated recursively in a conventional way as follows:

$$\begin{aligned}\hat{\boldsymbol{R}}_{x+}(1) &= \mathrm{diag}\{\sigma_{x_1}^2, \cdots, \sigma_{x_1}^2, \sigma_{x_2}^2, \cdots, \sigma_{x_2}^2, \\ &\quad \cdots, \sigma_{x_N}^2, \cdots, \sigma_{x_N}^2\}\,, \\ \hat{\boldsymbol{R}}_{x+}(k+1) &= \lambda\hat{\boldsymbol{R}}_{x+}(k) + \tilde{\boldsymbol{R}}_{x+}(k+1)\,, \\ &\quad \text{for } k \geq 1\,, \end{aligned} \tag{13.33}$$

where $\sigma_{x_n}^2$ $(n = 1, 2, \cdots, N)$ is the power of $x_n(k)$ and λ $(0 < \lambda < 1)$ is an exponential forgetting factor.

By using these approximations, we finally obtain an estimate $\boldsymbol{V}(k+1)$ for the mean Hessian matrix of $J(k+1)$

$$\begin{aligned}\boldsymbol{V}(k+1) = &2\hat{\boldsymbol{R}}_{x+}(k+1) - 4\hat{\boldsymbol{h}}(k)\hat{\boldsymbol{h}}^{\mathrm{T}}(k)\hat{\boldsymbol{R}}_{x+}(k+1) \\ &- 4\hat{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k)\hat{\boldsymbol{h}}^{\mathrm{T}}(k)\,, \end{aligned} \tag{13.34}$$

and hence deduce the unit-norm-constrained multichannel Newton algorithm [13.15]:

$$\begin{aligned}\hat{\boldsymbol{h}}(k+1) = &\{\hat{\boldsymbol{h}}(k) - 2\rho\boldsymbol{V}^{-1}(k+1)[\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) \\ &- \chi(k+1)\hat{\boldsymbol{h}}(k)]\} \\ &\Big/\{\|\hat{\boldsymbol{h}}(k) - 2\rho\boldsymbol{V}^{-1}(k+1)[\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) \\ &- \chi(k+1)\hat{\boldsymbol{h}}(k)]\|\}\,, \end{aligned} \tag{13.35}$$

where ρ is a new step size, close to but less than 1. The unit-norm-constrained MCN algorithm for blind SIMO identification is summarized in Table 13.2.

13.5 Unconstrained Multichannel LMS Algorithm with Optimal Step-Size Control

As explained in the previous sections, the constrained MCLMS algorithm converges slowly while the constrained MCN algorithm is computationally intensive. To improve performance, we can either find a way to accelerate the constrained MCLMS algorithm or reduce the complexity of the constrained MCN algorithm. We will develop algorithms along the former direction in this section and continue exploring this problem along the latter direction in the next section.

To accelerate the MCLMS algorithm, developing a scheme to determine the optimal step size during adaption is desirable and effective. We begin this endeavor with re-examining the update equation (13.15) or equivalently (13.24) of the unit-norm constrained MCLMS algorithm. As the adaptive algorithm proceeds, the cost function $J(k+1)$ diminishes and its gradient with respect to $\hat{\boldsymbol{h}}(k)$ can be approximated as

$$\nabla J(k+1) \approx \frac{2\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k)}{\|\hat{\boldsymbol{h}}(k)\|^2}\,. \tag{13.36}$$

At this point, removing the unit-norm constraint leads to a simplified, *unconstrained* MCLMS adaptive algorithm:

$$\hat{\boldsymbol{h}}(k+1) = \hat{\boldsymbol{h}}(k) - 2\mu\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k)\,, \tag{13.37}$$

which is theoretically equivalent to the adaptive algorithm proposed in [13.19] although the cost functions are defined in different ways in these two adaptive blind SIMO identification algorithms.

With such a simplified adaptive algorithm, the primary concern is whether it will converge to the trivial all-zero estimate. Fortunately this will not happen as long as the initial estimate $\hat{\boldsymbol{h}}(0)$ is not orthogonal to the true channel impulse response vector $\boldsymbol{h}$, as shown in [13.19]. This can easily be demonstrated by premultiplying (13.37) with $\boldsymbol{h}^{\mathrm{T}}$:

$$\boldsymbol{h}^{\mathrm{T}}\hat{\boldsymbol{h}}(k+1) = \boldsymbol{h}^{\mathrm{T}}\hat{\boldsymbol{h}}(k) - 2\mu\boldsymbol{h}^{\mathrm{T}}\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k)\,. \tag{13.38}$$

Table 13.3 The variable-step-size unconstrained multichannel LMS adaptive algorithm for the blind identification of an SIMO FIR system

Parameters:	$\hat{\boldsymbol{h}} = [\hat{\boldsymbol{h}}_1^{\mathrm{T}} \ \hat{\boldsymbol{h}}_2^{\mathrm{T}} \ \cdots \ \hat{\boldsymbol{h}}_N^{\mathrm{T}}]^{\mathrm{T}}$, model filter
Initialization:	$\hat{\boldsymbol{h}}_n(0) = [1 \ \ 1 \ \cdots \ 1]^{\mathrm{T}}$, $n = 1, 2, \cdots, N$
Computation:	For $k = 0, 1, 2, \cdots$, compute
(a)	$e_{ij}(k+1) = \boldsymbol{x}_i^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_j(k) - \boldsymbol{x}_j^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_i(k)$, $i, j = 1, 2, \cdots, N$
(b)	$\chi(k+1) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} e_{ij}^2(k+1)$
(c)	Construct the matrix $\tilde{\mathbf{R}}_{x+}(k+1)$ following Sect. 13.4.2
(d)	$\nabla J(k+1) \approx \dfrac{2\tilde{\mathbf{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k)}{\|\hat{\boldsymbol{h}}(k)\|^2}$
(e)	$\mu_{\mathrm{opt}}(k+1) = \dfrac{\hat{\boldsymbol{h}}^{\mathrm{T}}(k)\nabla J(k+1)}{\|\nabla J(k+1)\|^2}$
(f)	$\hat{\boldsymbol{h}}(k+1) = \hat{\boldsymbol{h}}(k) - \mu_{\mathrm{opt}}(k+1)\nabla J(k+1)$

Using the cross relation (13.4), we know that, in the absence of noise,

$$\boldsymbol{h}^{\mathrm{T}}\tilde{\boldsymbol{R}}_{x+}(k+1) = \boldsymbol{0}^{\mathrm{T}} . \tag{13.39}$$

This implies that the gradient $\nabla J(k+1)$ is orthogonal to $\boldsymbol{h}$ at any time k. As a result, (13.38) turns out to be

$$\boldsymbol{h}^{\mathrm{T}}\hat{\boldsymbol{h}}(k+1) = \boldsymbol{h}^{\mathrm{T}}\hat{\boldsymbol{h}}(k) . \tag{13.40}$$

This indicates that $\boldsymbol{h}^{\mathrm{T}}\hat{\boldsymbol{h}}(k)$ is time invariant for the unconstrained MCLMS algorithm. Provided that $\boldsymbol{h}^{\mathrm{T}}\hat{\boldsymbol{h}}(0) \neq 0$, $\hat{\boldsymbol{h}}(k)$ will not converge to zero.

Now we should feel comfortable to work on this unconstrained MCLMS algorithm and search for an optimal step size. Let us decompose the model filter $\hat{\boldsymbol{h}}(k)$ as follows:

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}_{\perp}(k) + \hat{\boldsymbol{h}}_{\parallel}(k) , \tag{13.41}$$

where $\hat{\boldsymbol{h}}_{\perp}(k)$ and $\hat{\boldsymbol{h}}_{\parallel}(k)$ are perpendicular and parallel to $\boldsymbol{h}$, respectively. Since the gradient $\nabla J(k+1)$ is orthogonal to $\boldsymbol{h}$ and $\boldsymbol{h}$ is parallel to $\boldsymbol{h}_{\parallel}(k)$, obviously $\nabla J(k+1)$ is orthogonal to $\boldsymbol{h}_{\parallel}(k)$ as well. Therefore, the update equation (13.37) of the unconstrained MCLMS algorithm can be decomposed into the following two separate equations:

$$\hat{\boldsymbol{h}}_{\perp}(k+1) = \hat{\boldsymbol{h}}_{\perp}(k) - \mu\nabla J(k+1) , \tag{13.42}$$

$$\hat{\boldsymbol{h}}_{\parallel}(k+1) = \hat{\boldsymbol{h}}_{\parallel}(k) . \tag{13.43}$$

From (13.42) and (13.43), it is clear that the unconstrained MCLMS algorithm adapts the model filter only in the direction perpendicular to $\boldsymbol{h}$. The component $\hat{\boldsymbol{h}}_{\parallel}(k)$ is not altered in the process of adaptation.

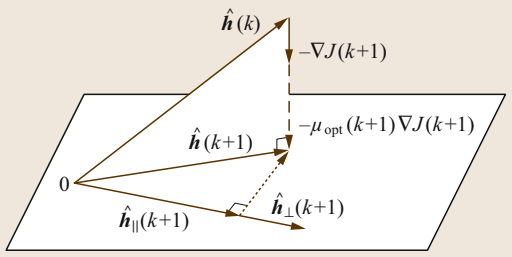


Fig. 13.2 Illustration of the optimal step size $\mu_{\mathrm{opt}}(k+1)$ for the unconstrained MCLMS blind SIMO identification algorithm in a three-dimensional space

As far as a general system identification algorithm is concerned, the most important performance measure should apparently be the difference between the true channel impulse response and the estimate. Using a blind multichannel identification method, the channel impulse responses of a SIMO FIR system can be estimated up to a scale. Therefore, the misalignment of an estimate $\hat{\boldsymbol{h}}(k)$ with respect to the true channel impulse response vector $\boldsymbol{h}$ would be:

$$d(k) = \min_{\alpha} \left\| \boldsymbol{h} - \alpha\hat{\boldsymbol{h}}(k) \right\|^2 , \tag{13.44}$$

where α is an arbitrary scale. Substituting (13.41) into (13.44) and finding the minimum yields

$$\begin{aligned} d(k) &= \min_{\alpha}[\|\hat{\boldsymbol{h}}(k)\|^2\alpha^2 - 2\|\hat{\boldsymbol{h}}_{\parallel}(k)\|\|\boldsymbol{h}\|\alpha + \|\boldsymbol{h}\|^2] \\ &= \frac{\|\boldsymbol{h}\|^2}{1 + (\|\hat{\boldsymbol{h}}_{\parallel}(k)\|/\|\hat{\boldsymbol{h}}_{\perp}(k)\|)^2} . \end{aligned} \tag{13.45}$$

Clearly the ratio of $\|\hat{\boldsymbol{h}}_{\parallel}(k)\|$ over $\|\hat{\boldsymbol{h}}_{\perp}(k)\|$ reflects how close the estimate is to the desired solution. With this feature in mind, the optimal step size $\mu_{\text{opt}}(k+1)$ for the unconstrained MCLMS algorithm at time $k+1$ would be the one that makes $\hat{\boldsymbol{h}}_{\perp}(k+1)$ have a minimum norm, i. e.,

$$\begin{aligned}\mu_{\text{opt}}(n+1) &= \arg\min_{\mu} \|\hat{\boldsymbol{h}}_{\perp}(k+1)\| \\ &= \arg\min_{\mu} \|\hat{\boldsymbol{h}}_{\perp}(k) - \mu\nabla J(k+1)\| \,.\end{aligned} \tag{13.46}$$

In order to minimize the norm of $\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(k) - \mu(k+1)\nabla J(k+1)$, as illustrated in Fig. 13.2, $\mu(k+1)$ should be chosen such that $\hat{\boldsymbol{h}}(k+1)$ is orthogonal to $\nabla J(k+1)$. Therefore, we project $\hat{\boldsymbol{h}}(k)$ onto $\nabla J(k+1)$ and obtain the optimal step size:

$$\mu_{\text{opt}}(k+1) = \frac{\hat{\boldsymbol{h}}^{\mathrm{T}}(k)\nabla J(k+1)}{\|\nabla J(k+1)\|^2} \,. \tag{13.47}$$

Finally, this new adaptive algorithm with the optimal step size is referred to as the variable-step-size unconstrained MCLMS (VSS-UMCLMS) [13.20] for blind SIMO identification and is summarized in Table 13.3.

13.6 Frequency-Domain Blind Multichannel Identification Algorithms

In this section, we study the problem of adaptive blind SIMO identification in the frequency domain. Time–frequency analysis utilizing the fast Fourier transform (FFT) is an important mathematical tool in signal processing. Since its introduction by *Dentino* et al. [13.21], adaptive filtering in the frequency domain has attracted a great deal of research interest and has recently become an essential constituent of adaptive filtering theory. By taking advantage of the computational efficiency of the FFT, a convolution of two signals can be quickly calculated. Moreover, a discrete Fourier transform processes a time sequence like a filter bank, which orthogonalizes the data, and therefore the coefficients of a frequency-domain adaptive filter can converge independently or even uniformly if the update is normalized properly. For single-channel cases, the derivation and implementation of a frequency-domain adaptive filter is relatively simple. However, when multiple channels are considered, the algorithmic complexity increases significantly with the number of channels.

Deriving a frequency-domain adaptive algorithm can involve a large number of variables in the form of both vectors and matrices, in both the time and frequency domains. Therefore, before formulating the problem and developing adaptive algorithms in the frequency domain, it is beneficial to clarify the notation used in the following. The notation is conventional in the time domain, but is specifically defined in the frequency domain. In the frequency domain, matrices and vectors are represented respectively by uppercase and lowercase bold, *italic* letters, and are further emphasized by an arrow underneath. However, for a vector the arrow is single-barred while for a matrix the arrow is double-barred. The difference in their appearance is illustrated by the following example:

- $\underset{\rightarrow}{\mathsf{x}}$ a vector in the frequency domain (sans serif, bold lowercase, with a single-headed single-barred arrow underneath),
- $\underset{\Rightarrow}{\mathsf{X}}$ a matrix in the frequency domain (sans serif, bold, uppercase, with a single-headed double-barred arrow underneath).

13.6.1 Frequency-Domain Multichannel LMS Algorithm

We begin with the definition of a signal $y_{ij}(k+1)$ by convolving the i-th channel output $x_i(k+1)$ with the j-th model filter $\hat{h}_j(k)$ $(i, j = 1, 2, \cdots, N)$:

$$y_{ij}(k+1) = x_i(k+1) * \hat{h}_j(k) \,. \tag{13.48}$$

As can be clearly seen in Sect. 13.4, $y_{ij}(k)$ is essential to calculate the error signal $e_{ij}(k+1)$ in (13.11), but involves intensive computation in the time domain. Here, we intend to perform digital filtering efficiently in the frequency domain using the overlap-save technique [13.22].

In the overlap-save technique, the signal is processed on a frame-by-frame basis. We are going to indicate the index of a frame with t. Let the vector $\tilde{\boldsymbol{y}}_{ij}(t+1)$ of length $2L$ denote the result of the circular convolution of $\boldsymbol{x}_i(t+1)$ and $\hat{\boldsymbol{h}}_j(t)$:

$$\tilde{\boldsymbol{y}}_{ij}(t+1) = \boldsymbol{C}_{x_i}(t+1)\hat{\boldsymbol{h}}_j^{10}(t) \,, \tag{13.49}$$

where

$$\tilde{\boldsymbol{y}}_{ij}(t+1) = [\tilde{y}_{ij}(tL)\ \ \tilde{y}_{ij}(tL+1) \cdots\ \tilde{y}_{ij}(tL+2L-1)]^{\mathrm{T}},$$

$$\boldsymbol{C}_{x_i}(t+1) = \begin{pmatrix} x_i(tL) & x_i(tL+2L-1) & \cdots & x_i(tL+1) \\ x_i(tL+1) & x_i(tL) & \cdots & x_i(tL+2) \\ \vdots & \vdots & \ddots & \vdots \\ x_i(tL+2L-1) & x_i(tL+2L-2) & \cdots & x_i(tL) \end{pmatrix},$$

$$\hat{\boldsymbol{h}}_j^{10}(t) = \left[\hat{\boldsymbol{h}}_j^{\mathrm{T}}(t)\ \ \boldsymbol{0}_{L\times 1}^{\mathrm{T}}\right]^{\mathrm{T}} = [\hat{h}_{j,0}(t)\ \cdots \hat{h}_{j,L-1}(t)\ \ 0\ \cdots\ 0]^{\mathrm{T}}.$$

Note that $\boldsymbol{C}_{x_i}(t+1)$ is a circulant matrix. It can easily be shown that the last L points in the circular convolution output, i. e., $\tilde{y}_{ij}(tL+L), \cdots, \tilde{y}_{ij}(tL+2L-1)$, are identical to the results of a linear convolution:

$$\begin{aligned} \boldsymbol{y}_{ij}(t+1) &= \boldsymbol{W}_{L\times 2L}^{01}\tilde{\boldsymbol{y}}_{ij}(t+1) \\ &= \boldsymbol{W}_{L\times 2L}^{01}\boldsymbol{C}_{x_i}(t+1)\hat{\boldsymbol{h}}_j^{10}(t) \\ &= \boldsymbol{W}_{L\times 2L}^{01}\boldsymbol{C}_{x_i}(t+1)\boldsymbol{W}_{2L\times L}^{10}\hat{\boldsymbol{h}}_j(t), \end{aligned} \tag{13.50}$$

where

$$\begin{aligned} \boldsymbol{y}_{ij}(t+1) &= [y_{ij}(tL)\ \ y_{ij}(tL+1) \cdots\ y_{ij}(tL+L-1)]^{\mathrm{T}}, \\ \boldsymbol{W}_{L\times 2L}^{01} &= (\boldsymbol{0}_{L\times L}\ \ \boldsymbol{I}_{L\times L}), \\ \boldsymbol{W}_{2L\times L}^{10} &= [\boldsymbol{I}_{L\times L}\ \ \boldsymbol{0}_{L\times L}]^{\mathrm{T}}, \\ \hat{\boldsymbol{h}}_j(t) &= [\hat{h}_{j,0}(t)\ \ \hat{h}_{j,1}(t)\ \cdots\ \hat{h}_{j,L-1}(t)]^{\mathrm{T}}. \end{aligned}$$

Therefore we overlap the input data blocks by L points. For each block of length L, we have $2L$ data inputs, discard the first L circular convolution results, and retain only the last L results as outputs.

With $\boldsymbol{y}_{ij}(t+1)$ defined, a block of the a priori error signal based on the cross-relation between the i-th and j-th channel outputs is determined as:

$$\begin{aligned} \boldsymbol{e}_{ij}(t+1) &= \boldsymbol{y}_{ij}(t+1) - \boldsymbol{y}_{ji}(t+1) \\ &= \boldsymbol{W}_{L\times 2L}^{01}\left[\boldsymbol{C}_{x_i}(t+1)\boldsymbol{W}_{2L\times L}^{10}\hat{\boldsymbol{h}}_j(t) - \boldsymbol{C}_{x_j}(t+1)\boldsymbol{W}_{2L\times L}^{10}\hat{\boldsymbol{h}}_i(t)\right]. \end{aligned} \tag{13.51}$$

Next, we use the FFT technique to perform a circular convolution efficiently. Let $\boldsymbol{F}_{L\times L}$ be the Fourier matrix of size $L\times L$:

$$\boldsymbol{F}_{L\times L} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \mathrm{e}^{-\mathrm{i}\frac{2\pi}{L}} & \mathrm{e}^{-\mathrm{i}\frac{4\pi}{L}} & \cdots & \mathrm{e}^{-\mathrm{i}\frac{2\pi(L-1)}{L}} \\ 1 & \mathrm{e}^{-\mathrm{i}\frac{4\pi}{L}} & \mathrm{e}^{-\mathrm{i}\frac{8\pi}{L}} & \cdots & \mathrm{e}^{-\mathrm{i}\frac{4\pi(L-1)}{L}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \mathrm{e}^{-\mathrm{i}\frac{2\pi(L-1)}{L}} & \mathrm{e}^{-\mathrm{i}\frac{4\pi(L-1)}{L}} & \cdots & \mathrm{e}^{-\mathrm{i}\frac{2\pi(L-1)^2}{L}} \end{pmatrix}, \tag{13.52}$$

where i is not the dummy variable we used above to indicate a system output, but stands for the square root of -1, i. e., $\mathrm{i} = \sqrt{-1}$. The matrix $\boldsymbol{F}_{L\times L}$ and its inverse are related by

$$\boldsymbol{F}_{L\times L}^{\mathrm{H}} = L\boldsymbol{F}_{L\times L}^{-1}, \tag{13.53}$$

where $(\cdot)^{\mathrm{H}}$ denotes the conjugate transpose or Hermitian transpose of a vector of a matrix. By using $\boldsymbol{F}_{2L\times 2L}$ and $\boldsymbol{F}_{2L\times 2L}^{-1}$, we can decompose the circulant matrix $\boldsymbol{C}_{x_i}(t+1)$:

$$\boldsymbol{C}_{x_i}(t+1) = \boldsymbol{F}_{2L\times 2L}^{-1}\underset{\Rightarrow x_i}{\mathsf{D}}(t+1)\boldsymbol{F}_{2L\times 2L}, \tag{13.54}$$

where $\underset{\Rightarrow x_i}{\mathsf{D}}(t+1)$ is a diagonal matrix whose diagonal elements are given by the DFT of the first column of $\boldsymbol{C}_{x_i}(t+1)$, i. e., the overlapped i-th channel output in the $(t+1)$-th block:

$$\boldsymbol{x}_i(t+1)_{2L\times 1} = [x_i(tL)\ \ x_i(tL+1) \cdots\ x_i(tL+2L-1)]^{\mathrm{T}}. \tag{13.55}$$

Now we multiply (13.51) by $\boldsymbol{F}_{L\times L}$ and use (13.54) to determine the block error sequence in the frequency domain:

$$\begin{aligned} \underset{\rightarrow ij}{\mathsf{e}}(t+1) &= \boldsymbol{F}_{L\times L}\boldsymbol{e}_{ij}(t+1) \\ &= \boldsymbol{F}_{L\times L}\boldsymbol{W}_{L\times 2L}^{01}\left[\boldsymbol{C}_{x_i}(t+1)\boldsymbol{W}_{2L\times L}^{10}\hat{\boldsymbol{h}}_j(t) - \boldsymbol{C}_{x_j}(t+1)\boldsymbol{W}_{2L\times L}^{10}\hat{\boldsymbol{h}}_i(t)\right] \\ &= \underset{\Rightarrow L\times 2L}{\mathsf{W}}^{01}\left[\underset{\Rightarrow x_i}{\mathsf{D}}(t+1)\underset{\Rightarrow 2L\times L}{\mathsf{W}}^{10}\underset{\rightarrow j}{\hat{\mathsf{h}}}(t) - \underset{\Rightarrow x_j}{\mathsf{D}}(t+1)\underset{\Rightarrow 2L\times L}{\mathsf{W}}^{10}\underset{\rightarrow i}{\hat{\mathsf{h}}}(t)\right], \end{aligned} \tag{13.56}$$

where

$$\begin{aligned} \underset{\Rightarrow L\times 2L}{\mathsf{W}}^{01} &= \boldsymbol{F}_{L\times L}\boldsymbol{W}_{L\times 2L}^{01}\boldsymbol{F}_{2L\times 2L}^{-1}, \\ \underset{\Rightarrow 2L\times L}{\mathsf{W}}^{10} &= \boldsymbol{F}_{2L\times 2L}\boldsymbol{W}_{2L\times L}^{10}\boldsymbol{F}_{L\times L}^{-1}, \end{aligned}$$

and $\underset{\rightarrow i}{\hat{\mathsf{h}}}(t)$ consists of the L-point DFTs of the vector $\hat{\boldsymbol{h}}_i(t)$ at the t-th block, i. e., $\underset{\rightarrow i}{\hat{\mathsf{h}}}(t) = \boldsymbol{F}_{L\times L}\hat{\boldsymbol{h}}_i(t)$.

Having derived a frequency-domain block error signal (13.56), we can consequently construct

a frequency-domain mean square error criterion analogous to its time-domain counterpart:

$$J_{\mathrm{f}} = E\{J_{\mathrm{f}}(t)\}\,, \tag{13.57}$$

where

$$J_{\mathrm{f}}(t) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \underset{\rightarrow ij}{\mathsf{e}}^{\mathrm{H}}(t)\, \underset{\rightarrow ij}{\mathsf{e}}(t)$$

is the instantaneous square error at the t-th block.

Taking the partial derivative of J_{f} with respect to $\underset{\rightarrow n}{\hat{\mathsf{h}}}^{*}(t)$ (where $n = 1, 2, \cdots, N$ and $(\cdot)^{*}$ stands for complex conjugate) and pretending that $\underset{\rightarrow n}{\hat{\mathsf{h}}}(t)$ is a constant [13.23] produces

$$\frac{\partial J_{\mathrm{f}}}{\partial \underset{\rightarrow n}{\hat{\mathsf{h}}}^{*}(t)} = E\left\{ \frac{\partial J_{\mathrm{f}}(t+1)}{\partial \underset{\rightarrow n}{\hat{\mathsf{h}}}^{*}(t)} \right\}\,. \tag{13.58}$$

In the LMS algorithm, the expectation is estimated with a one-point sample mean, i. e., the instantaneous value, which is given by:

$$\begin{aligned}
&\frac{\partial J_{\mathrm{f}}(t+1)}{\partial \underset{\rightarrow n}{\hat{\mathsf{h}}}^{*}(t)} \\
&= \frac{\partial}{\partial \underset{\rightarrow n}{\hat{\mathsf{h}}}^{*}(t)} \left[\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \underset{\rightarrow ij}{\mathsf{e}}^{\mathrm{H}}(t+1)\, \underset{\rightarrow ij}{\mathsf{e}}(t+1) \right] \\
&= \frac{\partial}{\partial \underset{\rightarrow n}{\hat{\mathsf{h}}}^{*}(t)} \left[\sum_{i=1}^{n-1} \underset{\rightarrow in}{\mathsf{e}}^{\mathrm{H}}(t+1)\, \underset{\rightarrow in}{\mathsf{e}}(t+1) \right] \\
&\quad + \frac{\partial}{\partial \underset{\rightarrow n}{\hat{\mathsf{h}}}^{*}(t)} \left[\sum_{j=n+1}^{N} \underset{\rightarrow nj}{\mathsf{e}}^{\mathrm{H}}(t+1)\, \underset{\rightarrow nj}{\mathsf{e}}(t+1) \right] \\
&= \sum_{i=1}^{n-1} \left[\underset{\Rightarrow L\times 2L}{\mathsf{W}}^{01} \underset{\Rightarrow x_i}{\mathsf{D}}(t+1) \underset{\Rightarrow 2L\times L}{\mathsf{W}}^{10} \right]^{\mathrm{H}} \underset{\rightarrow in}{\mathsf{e}}(t+1) \\
&\quad - \sum_{j=n+1}^{N} \left[\underset{\Rightarrow L\times 2L}{\mathsf{W}}^{01} \underset{\Rightarrow x_j}{\mathsf{D}}(t+1) \underset{\Rightarrow 2L\times L}{\mathsf{W}}^{10} \right]^{\mathrm{H}} \underset{\rightarrow nj}{\mathsf{e}}(t+1) \\
&= \sum_{i=1}^{N} \left[\underset{\Rightarrow L\times 2L}{\mathsf{W}}^{01} \underset{\Rightarrow x_i}{\mathsf{D}}(t+1) \underset{\Rightarrow 2L\times L}{\mathsf{W}}^{10} \right]^{\mathrm{H}} \underset{\rightarrow in}{\mathsf{e}}(t+1)\,,
\end{aligned} \tag{13.59}$$

where the last step follows from the fact $\underset{\rightarrow nn}{\mathsf{e}}(t) = \mathbf{0}$. Using this gradient, we obtain the frequency-domain *unconstrained* multichannel LMS algorithm:

$$\begin{aligned}
\underset{\rightarrow n}{\hat{\mathsf{h}}}(t+1) &= \underset{\rightarrow n}{\hat{\mathsf{h}}}(t) - \mu_{\mathrm{f}} \underset{\Rightarrow L\times 2L}{\mathsf{W}}^{10} \\
&\quad \cdot \sum_{i=1}^{N} \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\Rightarrow 2L\times L}{\mathsf{W}}^{01} \underset{\rightarrow in}{\mathsf{e}}(t+1)\,,
\end{aligned} \tag{13.60}$$

where μ_{f} is a small positive step size and

$$\begin{aligned}
\underset{\Rightarrow L\times 2L}{\mathsf{W}}^{10} &= \boldsymbol{F}_{L\times L}\, \boldsymbol{W}^{10}_{L\times 2L}\, \boldsymbol{F}^{-1}_{2L\times 2L} = \frac{1}{2} \left(\underset{\Rightarrow 2L\times L}{\mathsf{W}}^{10} \right)^{\mathrm{H}}\,, \\
\underset{\Rightarrow 2L\times L}{\mathsf{W}}^{01} &= \boldsymbol{F}_{2L\times 2L}\, \boldsymbol{W}^{01}_{2L\times L}\, \boldsymbol{F}^{-1}_{L\times L} = 2 \left(\underset{\Rightarrow L\times 2L}{\mathsf{W}}^{01} \right)^{\mathrm{H}}\,, \\
\boldsymbol{W}^{10}_{L\times 2L} &= (\boldsymbol{I}_{L\times L} \;\; \boldsymbol{0}_{L\times L})\,, \\
\boldsymbol{W}^{01}_{2L\times L} &= (\boldsymbol{0}_{L\times L} \;\; \boldsymbol{I}_{L\times L})^{\mathrm{T}}\,.
\end{aligned}$$

If the unit-norm constraint is employed, then we know that

$$\| \underset{\rightarrow}{\hat{\mathsf{h}}}(t) \|^2 = \frac{\|\hat{\boldsymbol{h}}(t)\|^2}{L} = \frac{1}{L}\,, \tag{13.61}$$

where

$$\underset{\rightarrow}{\hat{\mathsf{h}}}(t) = \left[\underset{\rightarrow 1}{\hat{\mathsf{h}}}^{\mathrm{T}}(t) \;\; \underset{\rightarrow 2}{\hat{\mathsf{h}}}^{\mathrm{T}}(t) \;\; \cdots \;\; \underset{\rightarrow N}{\hat{\mathsf{h}}}^{\mathrm{T}}(t) \right]^{\mathrm{T}}\,.$$

Enforcing the constraint on (13.60), we finally deduce the frequency-domain unit-norm-constrained multichannel LMS (FCMCLMS) algorithm [13.24]:

$$\begin{aligned}
&\underset{\rightarrow n}{\hat{\mathsf{h}}}(t+1) \\
&= \left\{ \underset{\rightarrow n}{\hat{\mathsf{h}}}(t) - \mu_{\mathrm{f}} \underset{\Rightarrow L\times 2L}{\mathsf{W}}^{10} \right. \\
&\quad \left. \cdot \sum_{i=1}^{N} \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\Rightarrow 2L\times L}{\mathsf{W}}^{01} \underset{\rightarrow in}{\mathsf{e}}(t+1) \right\} \\
&\quad \Big/ \left\{ \sqrt{L} \left\| \underset{\rightarrow n}{\hat{\mathsf{h}}}(t) - \mu_{\mathrm{f}} \underset{\Rightarrow L\times 2L}{\mathsf{W}}^{10} \right. \right. \\
&\quad \left. \left. \cdot \sum_{i=1}^{N} \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\Rightarrow 2L\times L}{\mathsf{W}}^{01} \underset{\rightarrow in}{\mathsf{e}}(t+1) \right\| \right\}\,, \\
&\quad n = 1, 2, \cdots, N\,.
\end{aligned} \tag{13.62}$$

This derivation is mathematically rigorous and the algorithm might look quite complicated to some readers since it involves a number of matrix multiplications. However, the implementation is not necessarily tedious. Actually many matrix multiplications are performed in the form of the Schur (element-by-element) product of two vectors. Here we want to briefly comment on the issue of how to skillfully and efficiently implement the

FCMCLMS algorithm. We will use (13.56) as an example, in which the error signal is calculated. The product $\underset{\Rightarrow 2L\times L}{\mathbf{W}}^{10}_{10}\ \underset{\rightarrow j}{\hat{\mathbf{h}}}(t)$ is obtained by appending L zeros to the end of $\hat{\boldsymbol{h}}_j(t)$ and taking the $2L$-point FFT:

$$\underset{\rightarrow j}{\hat{\mathbf{h}}}^{10}(t) = \underset{\Rightarrow 2L\times L}{\mathbf{W}}^{10}\ \underset{\rightarrow j}{\hat{\mathbf{h}}}(t) = \mathrm{FFT}_{2L}\{\hat{\boldsymbol{h}}_j^{10}(t)\}\,, \quad (13.63)$$

where $\mathrm{FFT}_{2L}\{\cdot\}$ denotes the $2L$-point fast Fourier transform and $\hat{\boldsymbol{h}}_j^{10}(t)$ is given in (13.49). Note that $\underset{\Rightarrow x_i}{\mathbf{D}}(t+1)$ is a diagonal matrix whose diagonal elements are the DFT coefficients of $\boldsymbol{x}_i(t+1)_{2L\times 1}$ given in (13.55), i. e.,

$$\begin{aligned}\underset{\Rightarrow x_i}{\mathbf{D}}(t+1) &= \mathrm{diag}[\underset{\rightarrow i}{\mathbf{x}}(t+1)_{2L\times 1}] \\ &= \mathrm{diag}[\mathrm{FFT}_{2L}\{\boldsymbol{x}_i(t+1)_{2L\times 1}\}]\,. \end{aligned} \quad (13.64)$$

Therefore multiplying $\underset{\Rightarrow x_i}{\mathbf{D}}(t+1)$ with $\underset{\Rightarrow 2L\times L}{\mathbf{W}}^{10}\ \underset{\rightarrow j}{\hat{\mathbf{h}}}(t)$ would be straightforward:

$$\begin{aligned}\underset{\Rightarrow x_i}{\mathbf{D}}(t+1)\underset{\Rightarrow 2L\times L}{\mathbf{W}}^{10}\ \underset{\rightarrow j}{\hat{\mathbf{h}}}(t) &= \underset{\Rightarrow x_i}{\mathbf{D}}(t+1)\underset{\rightarrow j}{\hat{\mathbf{h}}}^{10}(t) \\ &= \underset{\rightarrow i}{\mathbf{x}}(t+1)_{2L\times 1}\odot \underset{\rightarrow j}{\hat{\mathbf{h}}}^{10}(t)\,, \end{aligned} \quad (13.65)$$

where $\odot$ is the operator of the Schur product. Let $\boldsymbol{e}_{ij}(t+1)_{2L\times 1}$ be

$$\begin{aligned}\boldsymbol{e}_{ij}(t+1)_{2L\times 1} &= \mathrm{IFFT}_{2L}\Big\{\underset{\Rightarrow x_i}{\mathbf{D}}(t+1)\underset{\Rightarrow 2L\times L}{\mathbf{W}}^{10}\ \underset{\rightarrow j}{\hat{\mathbf{h}}}(t) \\ &\quad - \underset{\Rightarrow x_j}{\mathbf{D}}(t+1)\underset{\Rightarrow 2L\times L}{\mathbf{W}}^{10}\ \underset{\rightarrow i}{\hat{\mathbf{h}}}(t)\Big\} \\ &= \mathrm{IFFT}_{2L}\Big\{\underset{\rightarrow i}{\mathbf{x}}(t+1)_{2L\times 1}\odot \underset{\rightarrow j}{\hat{\mathbf{h}}}^{10}(t) \\ &\quad - \underset{\rightarrow j}{\mathbf{x}}(t+1)_{2L\times 1}\odot \underset{\rightarrow i}{\hat{\mathbf{h}}}^{10}(t)\Big\}\,, \end{aligned} \quad (13.66)$$

Table 13.4 The frequency-domain constrained multichannel LMS (FCMCLMS) adaptive algorithm for the blind identification of an SIMO FIR system

Parameters:	$\hat{\boldsymbol{h}} = [\hat{\boldsymbol{h}}_1^{\mathrm{T}}\ \hat{\boldsymbol{h}}_2^{\mathrm{T}}\ \cdots\ \hat{\boldsymbol{h}}_N^{\mathrm{T}}]^{\mathrm{T}}$, model filter
	$\mu_{\mathrm{f}} > 0$, step size
Initialization:	$\hat{\boldsymbol{h}}_n(0) = [1\ \ 0\ \ \cdots\ \ 0]^{\mathrm{T}}$, $n = 1, 2, \cdots, N$
	$\hat{\boldsymbol{h}}(0) = \hat{\boldsymbol{h}}(0)/\sqrt{N}$ (normalization)
Computation:	For $t = 0, 1, 2, \cdots,$ compute
(a)	$\underset{\rightarrow n}{\hat{\mathbf{h}}}^{10}(t) = \mathrm{FFT}_{2L}\{[\hat{\boldsymbol{h}}_n^{\mathrm{T}}(t)\ \ \mathbf{0}_{1\times L}]^{\mathrm{T}}\}$, $n = 1, 2, \cdots, N$
(b)	$\boldsymbol{x}_n(t+1)_{2L\times 1} = [x_n(tL)\ \ x_n(tL+1)\ \ \cdots\ \ x_n(tL+2L-1)]^{\mathrm{T}}$
(c)	$\underset{\rightarrow n}{\mathbf{x}}(t+1)_{2L\times 1} = \mathrm{FFT}_{2L}\{\boldsymbol{x}_n(t+1)_{2L\times 1}\}$
(d)	$\underset{\rightarrow ij}{\mathbf{e}}(t+1)_{2L\times 1} = \underset{\rightarrow i}{\mathbf{x}}(t+1)_{2L\times 1}\odot \underset{\rightarrow j}{\hat{\mathbf{h}}}^{10}(t) - \underset{\rightarrow j}{\mathbf{x}}(t+1)_{2L\times 1}\odot \underset{\rightarrow i}{\hat{\mathbf{h}}}^{10}(t)$, $i, j = 1, 2, \cdots, N$
(e)	$\boldsymbol{e}_{ij}(t+1)_{2L\times 1} = \mathrm{IFFT}_{2L}\{\underset{\rightarrow ij}{\mathbf{e}}(t+1)_{2L\times 1}\}$
(f)	Take the *last* L elements of $\boldsymbol{e}_{ij}(t+1)_{2L\times 1}$ to form $\boldsymbol{e}_{ij}(t+1)$
(g)	$\underset{\rightarrow ij}{\mathbf{e}}^{01}(t+1) = \mathrm{FFT}_{2L}\{[\mathbf{0}_{1\times L}\ \ \boldsymbol{e}_{ij}^{\mathrm{T}}(t+1)]^{\mathrm{T}}\}$
(h)	$\triangle\underset{\rightarrow n}{\hat{\mathbf{h}}}(t)_{2L\times 1} = \mu_{\mathrm{f}}\sum_{i=1}^{N}\underset{\rightarrow i}{\mathbf{x}}^{*}(t+1)_{2L\times 1}\odot \underset{\rightarrow in}{\mathbf{e}}^{01}(t+1)$
(i)	$\triangle\hat{\boldsymbol{h}}_n(t)_{2L\times 1} = \mathrm{IFFT}_{2L}\{\triangle\underset{\rightarrow n}{\hat{\mathbf{h}}}(t)_{2L\times 1}\}$
(j)	Take the *first* L elements of $\triangle\hat{\boldsymbol{h}}_n(t)_{2L\times 1}$ to form $\triangle\hat{\boldsymbol{h}}_n(t)$
(k)	$\hat{\boldsymbol{h}}_n(t+1) = \dfrac{\hat{\boldsymbol{h}}_n(t) - \triangle\hat{\boldsymbol{h}}_n(t)}{\left\|\hat{\boldsymbol{h}}_n(t) - \triangle\hat{\boldsymbol{h}}_n(t)\right\|}$, $n = 1, 2, \cdots, N$

where $\text{IFFT}_{2L}\{\cdot\}$ denotes $2L$-point *inverse* fast Fourier transform. Then

$$\boldsymbol{e}_{ij}(t+1) = \boldsymbol{W}^{01}_{L\times 2L}\boldsymbol{e}_{ij}(t+1)_{2L\times 1} \tag{13.67}$$

can be simply carried out by discarding the first L and keeping the last L elements of $\boldsymbol{e}_{ij}(t+1)_{2L\times 1}$ to construct $\boldsymbol{e}_{ij}(t+1)$. Finally $\underset{\rightarrow}{\mathsf{e}}_{ij}(t+1)$ is obtained by taking an L-point FFT of $\boldsymbol{e}_{ij}(\overline{t}+1)$.

The tricks developed for (13.56) can also be applied to (13.60) and (13.62), which will be left to the reader. The FCMCLMS algorithm is summarized in Table 13.4. Compared to the time-domain frame-based multichannel LMS algorithm, the frequency-domain implementation is computationally more efficient since it uses the FFT to calculate block convolution and block correlation in the frequency domain. For each processed frame, it can be checked from Table 13.4 that (N^2+2N) FFTs of $2L$ points are sufficient to implement exact linear convolutions between the channel outputs and the adaptive filter coefficients, and to implement exact correlations between the outputs and the error signals.

Having developed the frame-based MCLMS algorithm in the frequency domain, we need to demonstrate that it converges in the mean to the desired solution in a stationary environment. To establish this property, we begin by defining

$$\underset{\Rightarrow x_n}{\mathsf{S}}(t+1) = \mathsf{W}^{01}_{\Rightarrow L\times 2L}\,\underset{\Rightarrow x_n}{\mathsf{D}}(t+1)\mathsf{W}^{10}_{\Rightarrow 2L\times L}\,,$$
$$n = 1, 2, \cdots, N\,. \tag{13.68}$$

Then the block error signal in the frequency domain given by (13.56) can be rewritten as

$$\underset{\rightarrow ij}{\mathsf{e}}(t+1) = \underset{\Rightarrow x_i}{\mathsf{S}}(t+1)\underset{\rightarrow j}{\hat{\mathsf{h}}}(t) - \underset{\Rightarrow x_j}{\mathsf{S}}(t+1)\underset{\rightarrow i}{\hat{\mathsf{h}}}(t)\,, \tag{13.69}$$

and the instantaneous gradient (13.59) becomes

$$\frac{\partial J_{\text{f}}(t+1)}{\partial \underset{\rightarrow n}{\hat{\mathsf{h}}}^{*}(t)} = \sum_{i=1}^{N}\underset{\Rightarrow x_i}{\mathsf{S}}^{\text{H}}(t+1)\underset{\rightarrow in}{\mathsf{e}}(t+1)$$
$$= \underset{\Rightarrow}{\mathsf{S}}^{\text{H}}(t+1)\underset{\rightarrow n}{\mathsf{e}}(t+1)\,, \tag{13.70}$$

where

$$\underset{\Rightarrow}{\mathsf{S}}(t+1) = \left[\underset{\Rightarrow x_1}{\mathsf{S}}^{\text{H}}(t+1)\ \underset{\Rightarrow x_2}{\mathsf{S}}^{\text{H}}(t+1)\ \cdots\ \underset{\Rightarrow x_N}{\mathsf{S}}^{\text{H}}(t+1)\right]^{\text{H}},$$
$$\underset{\rightarrow n}{\mathsf{e}}(t+1) = \left[\underset{\rightarrow 1n}{\mathsf{e}}^{\text{T}}(t+1)\ \underset{\rightarrow 2n}{\mathsf{e}}^{\text{T}}(t+1)\ \cdots\ \underset{\rightarrow Nn}{\mathsf{e}}^{\text{T}}(t+1)\right]^{\text{T}}.$$

From (13.69), we then decompose the error signal $\underset{\rightarrow n}{\mathsf{e}}(t+1)$ associated with the n-th channel as follows:

$$\underset{\rightarrow n}{\mathsf{e}}(t+1) = \begin{pmatrix} \underset{\Rightarrow x_1}{\mathsf{S}}(t+1)\underset{\rightarrow n}{\hat{\mathsf{h}}}(t) - \underset{\Rightarrow x_n}{\mathsf{S}}(t+1)\underset{\rightarrow 1}{\hat{\mathsf{h}}}(t) \\ \underset{\Rightarrow x_2}{\mathsf{S}}(t+1)\underset{\rightarrow n}{\hat{\mathsf{h}}}(t) - \underset{\Rightarrow x_n}{\mathsf{S}}(t+1)\underset{\rightarrow 2}{\hat{\mathsf{h}}}(t) \\ \vdots \\ \underset{\Rightarrow x_N}{\mathsf{S}}(t+1)\underset{\rightarrow n}{\hat{\mathsf{h}}}(t) - \underset{\Rightarrow x_n}{\mathsf{S}}(t+1)\underset{\rightarrow N}{\hat{\mathsf{h}}}(t) \end{pmatrix}$$
$$= [\underset{\Rightarrow n}{\mathsf{U}}(t+1) - \underset{\Rightarrow n}{\mathsf{V}}(t+1)]\underset{\rightarrow}{\hat{\mathsf{h}}}(t)\,, \tag{13.71}$$

where

$$\underset{\Rightarrow n}{\mathsf{U}}(t+1) = \begin{pmatrix} \boldsymbol{0}_{L\times L} & \cdots & \boldsymbol{0}_{L\times L} & \underset{\times(t+1)}{\underset{\Rightarrow x_1}{\mathsf{S}}} & \boldsymbol{0}_{L\times L} & \cdots & \boldsymbol{0}_{L\times L} \\ \boldsymbol{0}_{L\times L} & \cdots & \boldsymbol{0}_{L\times L} & \underset{\times(t+1)}{\underset{\Rightarrow x_2}{\mathsf{S}}} & \boldsymbol{0}_{L\times L} & \cdots & \boldsymbol{0}_{L\times L} \\ \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ \underbrace{\boldsymbol{0}_{L\times L} \cdots \boldsymbol{0}_{L\times L}}_{n-1\text{ blocks}} & & & \underset{\times(t+1)}{\underset{\Rightarrow x_N}{\mathsf{S}}} & \underbrace{\boldsymbol{0}_{L\times L} \cdots \boldsymbol{0}_{L\times L}}_{N-n\text{ blocks}} & & \end{pmatrix}_{NL\times NL}$$
$$= (\boldsymbol{0}_{NL\times(n-1)L}\ \underset{\Rightarrow}{\mathsf{S}}(t+1)\ \boldsymbol{0}_{NL\times(N-n)L})_{NL\times NL}\,,$$
$$\underset{\Rightarrow n}{\mathsf{V}}(t+1) = \begin{pmatrix} \underset{\Rightarrow x_n}{\mathsf{S}}(t+1) & \boldsymbol{0}_{L\times L} & \cdots & \boldsymbol{0}_{L\times L} \\ \boldsymbol{0}_{L\times L} & \underset{\Rightarrow x_n}{\mathsf{S}}(t+1) & \cdots & \boldsymbol{0}_{L\times L} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{0}_{L\times L} & \boldsymbol{0}_{L\times L} & \cdots & \underset{\Rightarrow x_n}{\mathsf{S}}(t+1) \end{pmatrix}_{NL\times NL}.$$

Continuing, we can write

$$\underset{\Rightarrow}{\mathsf{S}}^{\text{H}}(t+1)\underset{\Rightarrow n}{\mathsf{U}}(t+1)$$
$$= \underset{\Rightarrow}{\mathsf{S}}^{\text{H}}(t+1)(\boldsymbol{0}_{NL\times(n-1)L}\ \underset{\Rightarrow}{\mathsf{S}}(t+1)\ \boldsymbol{0}_{NL\times(N-n)L})$$
$$= (\boldsymbol{0}_{NL\times(n-1)L}\ \textstyle\sum_{i=1}^{N}\underset{\Rightarrow ii}{\tilde{\mathsf{R}}}(t+1)\ \boldsymbol{0}_{NL\times(N-n)L})\,, \tag{13.72}$$

$$\underset{\Rightarrow}{\mathsf{S}}^{\text{H}}(t+1)\underset{\Rightarrow n}{\mathsf{V}}(t+1)$$
$$= (\underset{\Rightarrow 1n}{\tilde{\mathsf{R}}}(t+1)\ \underset{\Rightarrow 2n}{\tilde{\mathsf{R}}}(t+1)\ \cdots\ \underset{\Rightarrow Nn}{\tilde{\mathsf{R}}}(t+1))\,, \tag{13.73}$$

where

$$\underset{\Rightarrow ij}{\tilde{\mathsf{R}}}(t+1) = \underset{\Rightarrow x_i}{\mathsf{S}}^{\text{H}}(t+1)\underset{\Rightarrow x_j}{\mathsf{S}}(t+1)\,,$$
$$i, j = 1, 2, \cdots, N\,.$$

Incorporating (13.72) and (13.73) into (13.70) and (13.71) yields

$$\frac{\partial J_{\mathrm{f}}(t+1)}{\partial \hat{\underset{\rightarrow}{\mathsf{h}}}_n^*(t)} = \underset{\Rightarrow}{\mathsf{S}}^{\mathrm{H}}(t+1)[\underset{\Rightarrow n}{\mathsf{U}}(t+1) - \underset{\Rightarrow n}{\mathsf{V}}(t+1)]\hat{\underset{\rightarrow}{\mathsf{h}}}(t)$$
$$= \Big(-\underset{\Rightarrow 1n}{\tilde{\mathsf{R}}}(t+1) \;\; -\underset{\Rightarrow 2n}{\tilde{\mathsf{R}}}(t+1) \;\; \cdots \;\; \sum_{i\neq n}\underset{\Rightarrow ii}{\tilde{\mathsf{R}}}(t+1) \;\; \cdots \;\; -\underset{\Rightarrow Nn}{\tilde{\mathsf{R}}}(t+1)\Big)\hat{\underset{\rightarrow}{\mathsf{h}}}(t)\,. \tag{13.74}$$

Substituting this gradient estimate into (13.60) and concatenating the N impulse response vectors into a longer one produces a remarkably simple expression for the frequency-domain unconstrained MCLMS algorithm:

$$\hat{\underset{\rightarrow}{\mathsf{h}}}(t+1) = \hat{\underset{\rightarrow}{\mathsf{h}}}(t) - \mu_{\mathrm{f}}\underset{\Rightarrow}{\tilde{\mathsf{R}}}(t+1)\hat{\underset{\rightarrow}{\mathsf{h}}}(t)\,, \tag{13.75}$$

where

$$\underset{\Rightarrow}{\tilde{\mathsf{R}}}(t+1) = \begin{pmatrix} \sum_{i\neq 1}\underset{\Rightarrow ii}{\tilde{\mathsf{R}}}(t+1) & -\underset{\Rightarrow 21}{\tilde{\mathsf{R}}}(t+1) & \cdots & -\underset{\Rightarrow N1}{\tilde{\mathsf{R}}}(t+1) \\ -\underset{\Rightarrow 12}{\tilde{\mathsf{R}}}(t+1) & \sum_{i\neq 2}\underset{\Rightarrow ii}{\tilde{\mathsf{R}}}(t+1) & \cdots & -\underset{\Rightarrow N2}{\tilde{\mathsf{R}}}(t+1) \\ \vdots & \vdots & \ddots & \vdots \\ -\underset{\Rightarrow 1N}{\tilde{\mathsf{R}}}(t+1) & -\underset{\Rightarrow 2N}{\tilde{\mathsf{R}}}(t+1) & \cdots & \sum_{i\neq N}\underset{\Rightarrow ii}{\tilde{\mathsf{R}}}(t+1) \end{pmatrix}.$$

The update equation of the frequency-domain, unit-norm-constrained MCLMS algorithm is then expressed as

$$\hat{\underset{\rightarrow}{\mathsf{h}}}(t+1) = \frac{\hat{\underset{\rightarrow}{\mathsf{h}}}(t) - \mu_{\mathrm{f}}\underset{\Rightarrow}{\tilde{\mathsf{R}}}(t+1)\hat{\underset{\rightarrow}{\mathsf{h}}}(t)}{\sqrt{L}\left\|\hat{\underset{\rightarrow}{\mathsf{h}}}(t) - \mu_{\mathrm{f}}\underset{\Rightarrow}{\tilde{\mathsf{R}}}(t+1)\hat{\underset{\rightarrow}{\mathsf{h}}}(t)\right\|}\,. \tag{13.76}$$

Subtracting the true channel impulse responses $\underset{\rightarrow}{\mathsf{h}}$ from both sides of (13.75), we get the evolution of the misalignment in the frequency domain:

$$\delta\underset{\rightarrow}{\mathsf{h}}(t+1) = \delta\underset{\rightarrow}{\mathsf{h}}(t) - \mu_{\mathrm{f}}\underset{\Rightarrow}{\tilde{\mathsf{R}}}(t+1)\hat{\underset{\rightarrow}{\mathsf{h}}}(t)\,, \tag{13.77}$$

where

$$\delta\underset{\rightarrow}{\mathsf{h}}(t) = \hat{\underset{\rightarrow}{\mathsf{h}}}(t) - \underset{\rightarrow}{\mathsf{h}}\,.$$

Taking the expectation of (13.77) and invoking the independence assumption [13.17], i. e., that the channel output $\boldsymbol{x}_n(t+1)$, $n = 1, 2, \cdots, N$, and the filter coefficients $\hat{\underset{\rightarrow}{\mathsf{h}}}(t)$ are independent, we have

$$E\{\delta\underset{\rightarrow}{\mathsf{h}}(t+1)\} = E\{\delta\underset{\rightarrow}{\mathsf{h}}(t)\} - \mu_{\mathrm{f}}\underset{\Rightarrow}{\mathsf{R}}E\{\hat{\underset{\rightarrow}{\mathsf{h}}}(t)\}\,, \tag{13.78}$$

where

$$\underset{\Rightarrow}{\mathsf{R}} = E\{\underset{\Rightarrow}{\tilde{\mathsf{R}}}(t)\}\,.$$

Recall that the gradient of the mean-square error is zero for the desired solution, i. e.,

$$\frac{\partial J_{\mathrm{f}}}{\partial \underset{\rightarrow}{\mathsf{h}}} = \underset{\Rightarrow}{\mathsf{R}}\underset{\rightarrow}{\mathsf{h}} = \mathbf{0}\,,$$

using which (13.78) can be written as

$$E\{\delta\underset{\rightarrow}{\mathsf{h}}(t+1)\} = (\boldsymbol{I}_{NL\times NL} - \mu_{\mathrm{f}}\underset{\Rightarrow}{\mathsf{R}})E\{\delta\underset{\rightarrow}{\mathsf{h}}(t)\}\,. \tag{13.79}$$

Therefore, if the step size satisfies

$$0 < \mu_{\mathrm{f}} < \frac{2}{\lambda_{\max}}\,, \tag{13.80}$$

where $\lambda_{\max}$ is the largest eigenvalue of the matrix $\underset{\Rightarrow}{\mathsf{R}}$, then the misalignment mean $E\left\{\delta\underset{\rightarrow}{\mathsf{h}}(t)\right\}$ converges to zero and the estimated filter coefficients converge in the mean to the desired solutions.

13.6.2 Frequency-Domain Normalized Multichannel LMS Algorithm

The *unnormalized* frequency-domain MCLMS algorithm developed in the previous section is certainly computationally efficient. However, its convergence is still unsatisfactorily slow because of nonuniform convergence rates of the filter coefficients and cross-coupling between them. In this section, Newton's method will first be used and then necessary simplifications will be investigated to develop a frequency-domain *normalized* MCLMS algorithm for which the eigenvalue disparity is reduced and the convergence is accelerated.

To apply Newton's method, we need to evaluate the Hessian matrix of $J_{\mathrm{f}}(t+1)$ with respect to the filter coefficients, which can be computed by taking the row gradient of (13.70)

$$\underset{\Rightarrow n}{\mathsf{T}}(t+1) = \frac{\partial}{\partial \hat{\underset{\rightarrow}{\mathsf{h}}}_n^{\mathrm{T}}(t)}\left(\frac{\partial J_{\mathrm{f}}(t+1)}{\partial \hat{\underset{\rightarrow}{\mathsf{h}}}_n^*(t)}\right)$$
$$= \frac{\partial}{\partial \hat{\underset{\rightarrow}{\mathsf{h}}}_n^{\mathrm{T}}(t)}[\underset{\Rightarrow}{\mathsf{S}}^{\mathrm{H}}(t+1)\underset{\rightarrow n}{\mathsf{e}}(t+1)]$$
$$= \underset{\Rightarrow}{\mathsf{S}}^{\mathrm{H}}(t+1)\frac{\partial \underset{\rightarrow n}{\mathsf{e}}(t+1)}{\partial \hat{\underset{\rightarrow}{\mathsf{h}}}_n(t)}$$
$$= \sum_{i=1,i\neq n}^{N}\underset{\Rightarrow x_i}{\mathsf{S}}^{\mathrm{H}}(t+1)\underset{\Rightarrow x_i}{\mathsf{S}}(t+1)\,. \tag{13.81}$$

Then the filter coefficients will be updated as follows

$$\begin{aligned}\underset{\rightarrow n}{\hat{\mathsf{h}}}(t+1) &= \underset{\rightarrow n}{\hat{\mathsf{h}}}(t) - \rho_{\mathrm{f}} \underset{\Rightarrow n}{\mathsf{T}}^{-1}(t+1) \\ &\quad \times \underset{\Rightarrow}{\mathsf{S}}^{\mathrm{H}}(t+1) \underset{\rightarrow n}{\mathsf{e}}(t+1) \\ &= \underset{\rightarrow n}{\hat{\mathsf{h}}}(t) - \rho_{\mathrm{f}} \underset{\Rightarrow n}{\mathsf{T}}^{-1}(t+1) \underset{\Rightarrow L\times 2L}{\mathsf{W}^{10}} \\ &\quad \cdot \sum_{i=1}^{N} \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\Rightarrow 2L\times L}{\mathsf{W}^{01}} \underset{\rightarrow in}{\mathsf{e}}(t+1)\,, \end{aligned} \tag{13.82}$$

where ρ_{f} is a new step size. Multiplying (13.82) from the left by $\underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}}$ yields

$$\begin{aligned}\underset{\rightarrow n}{\hat{\mathsf{h}}}^{10}(t+1) &= \underset{\rightarrow n}{\hat{\mathsf{h}}}^{10}(t) - \rho_{\mathrm{f}} \underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}} \underset{\Rightarrow n}{\mathsf{T}}^{-1}(t+1) \underset{\Rightarrow L\times 2L}{\mathsf{W}^{10}} \\ &\quad \cdot \sum_{i=1}^{N} \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\rightarrow in}{\mathsf{e}}^{01}(t+1)\,, \end{aligned} \tag{13.83}$$

where the symbols

$$\begin{aligned}\underset{\rightarrow n}{\hat{\mathsf{h}}}^{10}(t) &= \underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}} \underset{\rightarrow n}{\hat{\mathsf{h}}}(t) = \boldsymbol{F}_{2L\times 2L}\left[\hat{\boldsymbol{h}}_n^{\mathrm{T}}(t)\ \ \boldsymbol{0}_{1\times L}\right]^{\mathrm{T}}, \\ \underset{\rightarrow in}{\mathsf{e}}^{01}(t+1) &= \underset{\Rightarrow 2L\times L}{\mathsf{W}^{01}} \underset{\rightarrow in}{\mathsf{e}}(t+1) \\ &= \boldsymbol{F}_{2L\times 2L}\left[\boldsymbol{0}_{1\times L}\ \ \boldsymbol{e}_{in}^{\mathrm{T}}(t+1)\right]^{\mathrm{T}}, \end{aligned}$$

are shown in Table 13.4.

The frequency-domain Newton algorithm is able to converge quickly, but unfortunately the matrix $\underset{\Rightarrow n}{\mathsf{T}}(t+1)$ is not diagonal and finding its inverse is computationally demanding, particularly when L is large. It is desirable to reduce the complexity by approximation, which leads to the normalized frequency-domain MCLMS algorithm [13.24].

Let us expand the product in (13.81) using the definition (13.68) to find:

$$\begin{aligned}&\underset{\Rightarrow x_i}{\mathsf{S}}^{\mathrm{H}}(t+1) \underset{\Rightarrow x_i}{\mathsf{S}}(t+1) \\ &= \underset{\Rightarrow L\times 2L}{\mathsf{W}^{10}} \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\Rightarrow 2L\times 2L}{\mathsf{W}^{01}} \underset{\Rightarrow x_i}{\mathsf{D}}(t+1) \underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}}\,, \end{aligned} \tag{13.84}$$

where

$$\begin{aligned}\underset{\Rightarrow 2L\times 2L}{\mathsf{W}^{01}} &= \underset{\Rightarrow 2L\times L}{\mathsf{W}^{01}} \underset{\Rightarrow L\times 2L}{\mathsf{W}^{01}} \\ &= \boldsymbol{F}_{2L\times 2L} \begin{pmatrix} \boldsymbol{0}_{L\times L} & \boldsymbol{0}_{L\times L} \\ \boldsymbol{0}_{L\times L} & \boldsymbol{I}_{L\times L} \end{pmatrix} \boldsymbol{F}_{2L\times 2L}^{-1}\,, \end{aligned}$$

whose elements on the main diagonal dominate, as shown in [13.25]. When L is large, $2\underset{\Rightarrow 2L\times 2L}{\mathsf{W}^{01}}$ can be well approximated by the identity matrix

$$2\underset{\Rightarrow 2L\times 2L}{\mathsf{W}^{01}} \approx \boldsymbol{I}_{2L\times 2L}\,. \tag{13.85}$$

Therefore, (13.84) becomes

$$\begin{aligned}&\underset{\Rightarrow x_i}{\mathsf{S}}^{\mathrm{H}}(t+1) \underset{\Rightarrow x_i}{\mathsf{S}}(t+1) \\ &\approx \frac{1}{2} \underset{\Rightarrow L\times 2L}{\mathsf{W}^{10}} \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\Rightarrow x_i}{\mathsf{D}}(t+1) \underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}}\,, \end{aligned} \tag{13.86}$$

and (13.81) becomes

$$\begin{aligned}\underset{\Rightarrow n}{\mathsf{T}}(t+1) &\approx \frac{1}{2} \underset{\Rightarrow L\times 2L}{\mathsf{W}^{10}} \left[\sum_{i=1, i\neq n}^{N} \underset{\Rightarrow x_i}{\mathsf{P}}(t+1)\right] \underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}} \\ &= \frac{1}{2} \underset{\Rightarrow L\times 2L}{\mathsf{W}^{10}} \underset{\Rightarrow \not{n}}{\mathsf{P}}(t+1) \underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}}\,, \end{aligned} \tag{13.87}$$

where

$$\begin{aligned}&\underset{\Rightarrow x_i}{\mathsf{P}}(t+1) = \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\Rightarrow x_i}{\mathsf{D}}(t+1)\,, \\ &\underset{\Rightarrow \not{n}}{\mathsf{P}}(t+1) = \sum_{i=1, i\neq n}^{N} \underset{\Rightarrow x_i}{\mathsf{P}}(t+1)\,, \quad n = 1, 2, \cdots, N\,, \end{aligned}$$

which are diagonal matrices.

There is a useful relation between the inverses of the $\underset{\Rightarrow n}{\mathsf{T}}(t+1)$ and $\underset{\Rightarrow \not{n}}{\mathsf{P}}(t+1)$ matrices given by [13.25]

$$\underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}} \underset{\Rightarrow n}{\mathsf{T}}^{-1}(t+1) \underset{\Rightarrow L\times 2L}{\mathsf{W}^{10}} = 2\underset{\Rightarrow 2L\times 2L}{\mathsf{W}^{10}} \underset{\Rightarrow \not{n}}{\mathsf{P}}^{-1}(t+1)\,, \tag{13.88}$$

where

$$\underset{\Rightarrow 2L\times 2L}{\mathsf{W}^{10}} = \underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}} \underset{\Rightarrow L\times 2L}{\mathsf{W}^{10}}\,.$$

This relation can be justified by post-multiplying both sides of the expression by $\underset{\Rightarrow \not{n}}{\mathsf{P}}(t+1) \underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}}$, using (13.87), and recognizing that

$$\underset{\Rightarrow 2L\times 2L}{\mathsf{W}^{10}} \underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}} = \underset{\Rightarrow 2L\times L}{\mathsf{W}^{10}}\,.$$

Substituting (13.88) into (13.83) produces

$$\begin{aligned}\underset{\rightarrow n}{\hat{\mathsf{h}}}^{10}(t+1) &= \underset{\rightarrow n}{\hat{\mathsf{h}}}^{10}(t) - 2\rho_{\mathrm{f}} \underset{\Rightarrow 2L\times 2L}{\mathsf{W}^{10}} \underset{\Rightarrow \not{n}}{\mathsf{P}}^{-1}(t+1) \\ &\quad \cdot \sum_{i=1}^{N} \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\rightarrow in}{\mathsf{e}}^{01}(t+1)\,. \end{aligned} \tag{13.89}$$

If the matrix $2\underset{\Rightarrow 2L\times 2L}{\mathsf{W}^{10}}$ is approximated by the identity matrix similar to (13.85), we finally deduce the the frequency-domain *normalized* MCLMS (FNMCLMS) algorithm [13.24]:

$$\begin{aligned}\underset{\rightarrow n}{\hat{\mathsf{h}}}^{10}(t+1) &= \underset{\rightarrow n}{\hat{\mathsf{h}}}^{10}(t) - \rho_{\mathrm{f}} \underset{\Rightarrow \not{n}}{\mathsf{P}}^{-1}(t+1) \\ &\quad \cdot \sum_{i=1}^{N} \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\rightarrow in}{\mathsf{e}}^{01}(t+1)\,, \\ &\quad n = 1, 2, \cdots, N\,. \end{aligned} \tag{13.90}$$

An interesting interpretation follows if we compare the frequency-domain normalized (13.90) and unnormalized (13.60) MCLMS algorithms. In the frequency-domain unnormalized MCLMS algorithm, the correction that is applied to $\underset{\rightarrow n}{\hat{\mathsf{h}}}(t)$ is approximately proportional to the power spectrum $\underset{\Rightarrow \eta'}{\mathsf{P}}(t+1)$ of the multiple channel outputs; this can be seen from (13.60) and the definition of $\underset{\Rightarrow \eta'}{\mathsf{P}}(t+1)$, making the approximation $2\underset{\Rightarrow 2L\times 2L}{\mathsf{W}}^{01} \approx \underset{2L\times 2L}{\boldsymbol{I}}\overset{\Rightarrow \eta'}{}$. When the magnitudes of the channel outputs are large, the gradient noise amplification can be evident. With the normalization of the correction by $\underset{\Rightarrow \eta'}{\mathsf{P}}(t+1)$ as performed in the normalized FMCLMS algorithm, this noise amplification problem is diminished and the variability of the convergence rates due to

Table 13.5 The frequency-domain normalized multichannel LMS (FNMCLMS) adaptive algorithm for the blind identification of an SIMO FIR system

Parameters:

$\hat{\boldsymbol{h}} = [\hat{\boldsymbol{h}}_1^{\mathrm{T}} \ \hat{\boldsymbol{h}}_2^{\mathrm{T}} \ \cdots \ \hat{\boldsymbol{h}}_N^{\mathrm{T}}]^{\mathrm{T}}$, model filter

$\rho_{\mathrm{f}} > 0$, step size

$\lambda_{\mathrm{p}} = [1 - 1/(3L)]^L$, forgetting factor

$\delta > 0$, regularization factor

Initialization:

$\hat{\boldsymbol{h}}_n(0) = [1 \ 0 \ \cdots \ 0]^{\mathrm{T}}, \quad n = 1, 2, \cdots, N$

$\hat{\boldsymbol{h}}(0) = \hat{\boldsymbol{h}}(0)/\sqrt{N}$ (normalization)

Computation:

For $t = 0, 1, 2, \cdots$, compute

(a) $\underset{\rightarrow n}{\hat{\mathsf{h}}}^{10}(t) = \mathrm{FFT}_{2L}\{[\hat{\boldsymbol{h}}_n^{\mathrm{T}}(t) \ \boldsymbol{0}_{1\times L}]^{\mathrm{T}}\}, \quad n = 1, 2, \cdots, N$

(b) $\boldsymbol{x}_n(t+1)_{2L\times 1} = [x_n(tL) \ x_n(tL+1) \ \cdots \ x_n(tL+2L-1)]^{\mathrm{T}}$

(c) $\underset{\rightarrow n}{\mathsf{x}}(t+1)_{2L\times 1} = \mathrm{FFT}_{2L}[\boldsymbol{x}_n(t+1)_{2L\times 1}]$

(d) $\underset{\rightarrow ij}{\mathsf{e}}(t+1)_{2L\times 1} = \underset{\rightarrow i}{\mathsf{x}}(t+1)_{2L\times 1} \odot \underset{\rightarrow j}{\hat{\mathsf{h}}}^{10}(t) - \underset{\rightarrow j}{\mathsf{x}}(t+1)_{2L\times 1} \odot \underset{\rightarrow i}{\hat{\mathsf{h}}}^{10}(t)$,
$i, j = 1, 2, \cdots, N$

(e) $\boldsymbol{e}_{ij}(t+1)_{2L\times 1} = \mathrm{IFFT}_{2L}\{\underset{\rightarrow ij}{\mathsf{e}}(t+1)_{2L\times 1}\}$

(f) Take the *last* L elements of $\boldsymbol{e}_{ij}(t+1)_{2L\times 1}$ to form $\boldsymbol{e}_{ij}(t+1)$

(g) $\underset{\rightarrow ij}{\mathsf{e}}^{01}(t+1) = \mathrm{FFT}_{2L}\{[\boldsymbol{0}_{1\times L} \ \boldsymbol{e}_{ij}^{\mathrm{T}}(t+1)]^{\mathrm{T}}\}$

(h) $\underset{\rightarrow \eta'}{\tilde{\mathsf{p}}}(t+1) = \sum_{i=1, i\neq n}^{N} \underset{\rightarrow i}{\mathsf{x}}^*(t+1)_{2L\times 1} \odot \underset{\rightarrow i}{\mathsf{x}}(t+1)_{2L\times 1}$

(i) $\underset{\rightarrow \eta'}{\mathsf{p}}(t+1) = \begin{cases} \underset{\rightarrow \eta'}{\tilde{\mathsf{p}}}(t+1), & t = 0 \\ \lambda_{\mathrm{p}} \underset{\rightarrow \eta'}{\mathsf{p}}(t) + (1-\lambda_{\mathrm{p}}) \underset{\rightarrow \eta'}{\tilde{\mathsf{p}}}(t+1), & t > 0 \end{cases}$

(j) Take the element-by-element reciprocal of $[\underset{\rightarrow \eta'}{\mathsf{p}}(t+1) + \delta \boldsymbol{1}_{2L\times 1}]$ to form $\underset{\rightarrow \eta'}{\mathsf{p}}^{-1}(t+1)$

(k) $\Delta\underset{\rightarrow n}{\hat{\mathsf{h}}}(t)_{2L\times 1} = \rho_{\mathrm{f}} \underset{\rightarrow \eta'}{\mathsf{p}}^{-1}(t+1) \odot \left[\sum_{i=1}^{N} \underset{\rightarrow i}{\mathsf{x}}^*(t+1)_{2L\times 1} \odot \underset{\rightarrow in}{\mathsf{e}}^{01}(t+1)\right]$

(l) $\Delta\hat{\boldsymbol{h}}_n(t)_{2L\times 1} = \mathrm{IFFT}_{2L}\{\Delta\underset{\rightarrow n}{\hat{\mathsf{h}}}(t)_{2L\times 1}\}$

(m) Take the *first* L elements of $\Delta\hat{\boldsymbol{h}}_n(t)_{2L\times 1}$ to form $\Delta\hat{\boldsymbol{h}}_n(t)$

(n) $\hat{\boldsymbol{h}}_n(t+1) = \dfrac{\hat{\boldsymbol{h}}_n(t) - \Delta\hat{\boldsymbol{h}}_n(t)}{\left\|\hat{\boldsymbol{h}}_n(t) - \Delta\hat{\boldsymbol{h}}_n(t)\right\|}, \quad n = 1, 2, \cdots, N$

the change of signal level is eliminated. In order to estimate a more-stable power spectrum, a recursive scheme is employed in practice [13.24]:

$$\underset{\Rightarrow y}{\mathsf{P}}(t+1) = \lambda_{\mathrm{p}} \underset{\Rightarrow y}{\mathsf{P}}(t) + (1-\lambda_{\mathrm{p}}) \cdot \sum_{i=1, i\neq n}^{N} \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\Rightarrow x_i}{\mathsf{D}}(t+1) \,, \quad n = 1, 2, \cdots, N \,, \tag{13.91}$$

where λ_{p} is a forgetting factor that may appropriately be set as

$$\lambda_{\mathrm{p}} = \left(1 - \frac{1}{3L}\right)^{L}$$

for the FNMCLMS algorithm. Although the FNMCLMS algorithm overcomes the problem of noise amplification, we are now faced with a similar problem that occurs when the channel outputs becomes too small. An alternative approach, therefore, is to insert a positive number δ into the normalization, which leads to the following modification to the FNMCLMS algorithm:

$$\underset{\rightarrow n}{\hat{\mathsf{h}}}^{10}(t+1) = \underset{\rightarrow n}{\hat{\mathsf{h}}}^{10}(t) - \rho_{\mathrm{f}} [\underset{\Rightarrow y}{\mathsf{P}}(t+1) + \delta \boldsymbol{I}_{2L\times 2L}]^{-1} \cdot \sum_{i=1}^{N} \underset{\Rightarrow x_i}{\mathsf{D}}^{*}(t+1) \underset{\rightarrow in}{\mathsf{e}}^{01}(t+1) \,, \quad n = 1, 2, \cdots, N \,. \tag{13.92}$$

From a computational point of view, the modified FNMCLMS algorithm can easily be implemented even for a real-time application since the normalization matrix $\underset{\Rightarrow y}{\mathsf{P}}(t+1) + \delta \boldsymbol{I}_{2L\times 2L}$ is diagonal and it is straightforward to find its inverse.

The FNMCLMS algorithm is summarized in Table 13.5.

13.7 Adaptive Multichannel Exponentiated Gradient Algorithm

We have developed in the previous Sections a number of adaptive algorithms for blind identification of SIMO FIR systems. In fact, all these algorithms are based on the classical stochastic gradient and their update rules do not discriminate against the model filter coefficients with significantly distinct magnitudes. Intuitively, however, it seems possible to develop a better adaptive blind SIMO identification algorithm by taking advantage of the sparseness in acoustic impulse responses.

It has only very recently been recognized that the sparseness of an impulse response can be exploited in adaptive algorithms to accelerate their initial convergence and improve their tracking performance. The proportionate normalized LMS (PNLMS) proposed by *Duttweiler* [13.26] was perhaps one of the first such algorithms, which was introduced for network echo cancelation. It was shown in [13.27] that the PNLMS is an approximation of the so-called exponentiated gradient algorithm with positive and negative weights (EG± algorithm). EG± was proposed by *Kivinen* and *Warmuth* in the context of computational learning theory [13.28]. In [13.29], a general expression of the mean-squared error (MSE) was derived for the EG± algorithm, illustrating that for sparse impulse responses, the EG± algorithm, like the PNLMS, converges more quickly than the LMS for a given asymptotic MSE. Both the PNLMS and EG± algorithms were developed for identifying a single channel with reference signals [13.27]. In this section, we will investigate how to apply the concept of exponentiated gradient for blind identification of sparse acoustic SIMO systems and review the multichannel EG± (MCEG±) algorithm [13.30].

We begin with the introduction of a new way to develop adaptive algorithms for blind SIMO identification, which is different from the gradient descent perspective described in Sect. 13.4. In addition to the unconstrained, a priori error signal (13.11), we define an a posteriori error:

$$\varepsilon_{ij}(k+1) = \boldsymbol{x}_i^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_j(k+1) - \boldsymbol{x}_j^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_j(k+1) \,, \quad i, j = 1, 2, \cdots, N \,. \tag{13.93}$$

Then an adaptive algorithm that adjusts the new model filter $\hat{\boldsymbol{h}}(k+1)$ from the old one $\hat{\boldsymbol{h}}(k)$ can be derived by minimizing the function

$$J[\hat{\boldsymbol{h}}(k+1)] = d[\hat{\boldsymbol{h}}(k+1), \hat{\boldsymbol{h}}(k)] + \eta \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \varepsilon_{ij}^{2}(k+1) \,, \tag{13.94}$$

where $d[\hat{\boldsymbol{h}}(k+1), \hat{\boldsymbol{h}}(k)]$ is a measure of the distance from the old to the new model filter, and η is a positive constant. The magnitude of η represents the importance of

correctness compared to the importance of conservativeness [13.28]. If η is very small, minimizing $J[\hat{\boldsymbol{h}}(k+1)]$ is very similar to minimizing $d[\hat{\boldsymbol{h}}(k+1), \hat{\boldsymbol{h}}(k)]$, so that the algorithm makes very small updates. On the other hand, if η is very large, the minimization of $J[\hat{\boldsymbol{h}}(k+1)]$ is almost equivalent to minimizing $d[\hat{\boldsymbol{h}}(k+1), \hat{\boldsymbol{h}}(k)]$ subject to the constraint $\varepsilon_{ij}(k+1)=0, \forall i, j$.

To minimize $J[\hat{\boldsymbol{h}}(k+1)]$, we need to set the partial derivative $\partial J[\hat{\boldsymbol{h}}(k+1)]/\partial\hat{\boldsymbol{h}}(k+1)$ to zero. Hence, the vector of channel impulse responses $\hat{\boldsymbol{h}}(k+1)$ will be found by solving

$$\frac{\partial d[\hat{\boldsymbol{h}}(k+1), \hat{\boldsymbol{h}}(k)]}{\partial\hat{\boldsymbol{h}}(k+1)} + \eta\frac{\partial}{\partial\hat{\boldsymbol{h}}(k+1)}\left[\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\varepsilon_{ij}^2(k+1)\right] = \boldsymbol{0}\,. \quad (13.95)$$

Similar to (13.22), it can be shown that

$$\frac{\partial}{\partial\hat{\boldsymbol{h}}(k+1)}\left[\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\varepsilon_{ij}^2(k+1)\right] = 2\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k+1)\,. \quad (13.96)$$

Then (13.95) becomes

$$\frac{\partial d[\hat{\boldsymbol{h}}(k+1), \hat{\boldsymbol{h}}(k)]}{\partial\hat{\boldsymbol{h}}(k+1)} + 2\eta\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k+1) = \boldsymbol{0}\,. \quad (13.97)$$

Solving (13.97) is in general very difficult. However, if the new weight vector $\hat{\boldsymbol{h}}(k+1)$ is close to the old weight vector $\hat{\boldsymbol{h}}(k)$, replacing $\hat{\boldsymbol{h}}(k+1)$ in the second term on the left-hand side of (13.97) with $\hat{\boldsymbol{h}}(k)$ is a reasonable approximation and the resulting equation

$$\frac{\partial d[\hat{\boldsymbol{h}}(k+1), \hat{\boldsymbol{h}}(k)]}{\partial\hat{\boldsymbol{h}}(k+1)} + 2\eta\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) = \boldsymbol{0} \quad (13.98)$$

is much easier to solve for all distance measures d.

The unconstrained multichannel LMS algorithm is easily obtained from (13.98) by using the squared Euclidean distance

$$d_{\mathrm{E}}[\hat{\boldsymbol{h}}(k+1), \hat{\boldsymbol{h}}(k)] = \left\|\hat{\boldsymbol{h}}(k+1) - \hat{\boldsymbol{h}}(k)\right\|_2^2\,. \quad (13.99)$$

If the unconstrained a posteriori error in (13.94) is replaced by a *constrained* a posteriori error signal, then the constrained multichannel LMS algorithm (13.24) can be deduced.

The exponentiated gradient (EG) algorithm with *positive* weights results from using the *relative entropy*, also known as the *Kullback–Leibler divergence*, for d:

$$d_{\mathrm{re}}[\hat{\boldsymbol{h}}(k+1), \hat{\boldsymbol{h}}(k)] = \sum_{l=0}^{NL-1}\hat{h}_l(k+1)\ln\frac{\hat{h}_l(k+1)}{\hat{h}_l(k)}\,, \quad (13.100)$$

with the constraint $\sum_{l=0}^{NL-1}\hat{h}_l(k+1)=u$, where $\hat{h}_l(k+1)$ is the l-th ($l=0, 1, \cdots, NL-1$) tap of the model filter $\hat{\boldsymbol{h}}(k+1)$ at time $k+1$, and u is a positive constant (note that this is different from the unit-norm constraint used in the constrained multichannel LMS algorithm). We slightly modify the function (13.94) as follows

$$J[\hat{\boldsymbol{h}}(k+1)] = d[\hat{\boldsymbol{h}}(k+1), \hat{\boldsymbol{h}}(k)] + \frac{\eta}{u}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\varepsilon_{ij}^2(k+1)\,. \quad (13.101)$$

As a result, (13.98) becomes

$$\frac{\partial d_{\mathrm{re}}[\hat{\boldsymbol{h}}(k+1), \hat{\boldsymbol{h}}(k)]}{\partial\hat{\boldsymbol{h}}(k+1)} + \frac{2\eta}{u}\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) + \kappa\boldsymbol{1} = \boldsymbol{0}\,, \quad (13.102)$$

where κ is a Lagrange multiplier and $\boldsymbol{1}=(1\ 1\ \cdots\ 1)^{\mathrm{T}}$ is a vector of ones. Substituting (13.100) into (13.102) and solving for $\hat{\boldsymbol{h}}(k+1)$ leads to the multichannel EG algorithm with positive weights (MCEG+):

$$\hat{h}_l(k+1) = u\frac{\hat{h}_l(k)r_l(n+1)}{\sum_{i=0}^{NL-1}\hat{h}_i(k)r_i(k+1)}, \quad l=0, 1, \cdots, NL-1\,, \quad (13.103)$$

where

$$r_l(k+1) = \exp\left[-\frac{2\eta}{u}g_l(k+1)\right]\,,$$

and $g_l(k+1)$ is the l-th element of the gradient vector

$$\boldsymbol{g}(k+1) = \tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k)\,. \quad (13.104)$$

We can clearly see the motivation for the EG algorithms by ignoring the normalization in (13.103) and taking the logarithm, which shows that the log weights use the same update as the MCLMS algorithm. Alternatively, this can be interpreted as exponentiating the update, hence the name EG. This has the effect of assigning larger relative updates to larger weights, thereby deemphasizing the effect of smaller weights. This is qualitatively similar to the idea that leads to the PNLMS algorithm, which makes the update *proportional* to the magnitude of the weight. This type of behavior is desirable for sparse impulse responses where small weights

do not contribute significantly to the *mean* solution but introduce an undesirable noise-like *variance*.

Since the MCEG+ algorithm works only for SIMO systems with positive channel impulse responses, it is useless for blind identification of an acoustic SIMO system. For an acoustic SIMO system with both positive and negative filter coefficients, we can decompose the model filter $\hat{\boldsymbol{h}}(k)$ into two components $\hat{\boldsymbol{h}}^+(k)$ and $\hat{\boldsymbol{h}}^-(k)$ (both with positive coefficients) such that $\hat{\boldsymbol{h}}(k)=\hat{\boldsymbol{h}}^+(k)-\hat{\boldsymbol{h}}^-(k)$. Then the function (13.101) becomes:

$$\begin{aligned}&J[\hat{\boldsymbol{h}}^+(k+1),\hat{\boldsymbol{h}}^-(k+1)]\\&=d_{\mathrm{re}}[\hat{\boldsymbol{h}}^+(k+1),\hat{\boldsymbol{h}}^+(k)]\\&\quad+d_{\mathrm{re}}[\hat{\boldsymbol{h}}^-(k+1),\hat{\boldsymbol{h}}^-(k)]\\&\quad+\frac{\eta}{u_1+u_2}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\varepsilon_{ij}^2(k+1)\,,\end{aligned}\tag{13.105}$$

where u_1 and u_2 are two positive constants. Since $\hat{\boldsymbol{h}}(k+1)=\boldsymbol{0}$ is an undesired solution, it is necessary to ensure that $\hat{\boldsymbol{h}}^+(k+1)$ and $\hat{\boldsymbol{h}}^-(k+1)$ are not equal to each other on initialization and throughout the process of adaptation. Among the many methods that can be used to do this, we introduce the following constraint proposed in [13.30]:

$$\sum_{l=0}^{NL-1}\hat{h}_l^+(k+1)=u_1\,,\qquad\sum_{l=0}^{NL-1}\hat{h}_l^-(k+1)=u_2\,,\tag{13.106}$$

where $u_1\neq u_2$. Using this constraint and taking derivatives of (13.105) with respect to $\hat{\boldsymbol{h}}^+(k+1)$ and $\hat{\boldsymbol{h}}^-(k+1)$ respectively produces:

$$\ln\frac{\hat{h}_l^+(k+1)}{\hat{h}_l^+(k)}+1+\frac{2\eta}{u_1+u_2}g_l(k+1)+\kappa_1=0\,,\tag{13.107}$$

$$\ln\frac{\hat{h}_l^-(k+1)}{\hat{h}_l^-(k)}+1-\frac{2\eta}{u_1+u_2}g_l(k+1)+\kappa_2=0\,,\tag{13.108}$$

where κ_1 and κ_2 are two Lagrange multipliers. Solving (13.107) and (13.108) for $\hat{h}_l^+(k+1)$ and $\hat{h}_l^-(k+1)$,

Table 13.6 The multichannel exponentiated gradient algorithm with positive and negative weights (MCEG±) for the blind identification of a sparse SIMO FIR system

Parameters:	$\hat{\boldsymbol{h}}=[\hat{\boldsymbol{h}}_1^{\mathrm{T}}\ \hat{\boldsymbol{h}}_2^{\mathrm{T}}\ \cdots\ \hat{\boldsymbol{h}}_N^{\mathrm{T}}]^{\mathrm{T}}=\hat{\boldsymbol{h}}^+-\hat{\boldsymbol{h}}^-$, model filter
	$\eta>0$, positive constant to balance correctness and conservativeness
	$u_1>0\,,u_2>0\,,$ and $u_1\neq u_2$
Initialization:	$\hat{\boldsymbol{h}}_n^+(0)=[u_1/N\ \ 0\ \cdots\ 0]^{\mathrm{T}}\quad n=1,2,\cdots,N$
	$\hat{\boldsymbol{h}}_n^-(0)=[u_2/N\ \ 0\ \cdots\ 0]^{\mathrm{T}}$
	$\hat{\boldsymbol{h}}(0)=\hat{\boldsymbol{h}}^+(0)-\hat{\boldsymbol{h}}^-(0)$
Computation:	For $k=0,1,2,\cdots,$ compute
	(a) Construct the matrix $\tilde{\mathbf{R}}_{x+}(k+1)$ following Sect. 13.4.2
	(b) $\boldsymbol{g}(k+1)=\tilde{\mathbf{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k)$
	(c) $r_l^+(k+1)=\exp\left[-\frac{2\eta}{u_1+u_2}g_l(k+1)\right]$
	$r_l^-(k+1)=\frac{1}{r_l^+(k+1)}\,,\quad l=0,1,\cdots,NL-1$
	(d) $\hat{\boldsymbol{h}}^+(k+1)=\frac{u_1[\hat{\boldsymbol{h}}^+(k)\odot\boldsymbol{r}^+(k+1)]}{[\hat{\boldsymbol{h}}^+(k)]^{\mathrm{T}}\boldsymbol{r}^+(k+1)}$
	$\hat{\boldsymbol{h}}^-(k+1)=\frac{u_2[\hat{\boldsymbol{h}}^-(k)\odot\boldsymbol{r}^-(k+1)]}{\left[\hat{\boldsymbol{h}}^-(k)\right]^{\mathrm{T}}\boldsymbol{r}^-(k+1)}$
	(e) $\hat{\boldsymbol{h}}(k+1)=\hat{\boldsymbol{h}}^+(k+1)-\hat{\boldsymbol{h}}^-(k+1)$

respectively, leads to the MCEG± algorithm:

$$\hat{h}_l^+(k+1) = u_1 \frac{\hat{h}_l^+(k) r_l^+(k+1)}{\sum_{i=0}^{NL-1} \hat{h}_i^+(k) r_i^+(k+1)} , \quad (13.109)$$

$$\hat{h}_l^-(k+1) = u_2 \frac{\hat{h}_l^-(k) r_l^-(k+1)}{\sum_{i=0}^{NL-1} \hat{h}_i^-(k) r_i^-(k+1)} , \quad (13.110)$$

where

$$r_l^+(k+1) = \exp\left[-\frac{2\eta}{u_1+u_2} g_l(k+1) \right] ,$$

$$r_l^-(k+1) = \exp\left[\frac{2\eta}{u_1+u_2} g_l(k+1) \right] = \frac{1}{r_l^+(k+1)} ,$$

which is summarized in Table 13.6.

Before wrapping up the development of the MCEG± algorithm, we would like to comment briefly on how to choose u_1 and u_2 in (13.106) properly. The constraint looks simple but, in practice, determining a proper set of u_1 and u_2 is actually quite difficult. Let $\boldsymbol{h}^{+0}$ be the non-negative component of $\boldsymbol{h}$, i. e.,

$$h_l^{+0} = \begin{cases} h_l, & h_l \geq 0 \\ 0, & h_l < 0 \end{cases} , \quad l = 0, 1, \cdots, NL-1 , \quad (13.111)$$

and

$$\boldsymbol{h}^{-0} = \boldsymbol{h}^{+0} - \boldsymbol{h} . \quad (13.112)$$

Then a decomposition of $\boldsymbol{h}$ can be expressed as

$$\boldsymbol{h}^+ = \boldsymbol{h}^{+0} + \boldsymbol{c} , \quad \boldsymbol{h}^- = \boldsymbol{h}^{-0} + \boldsymbol{c} , \quad (13.113)$$

where $\boldsymbol{c}$ is a constant vector. Since the coefficients of $\boldsymbol{h}^+$ and $\boldsymbol{h}^-$ are non-negative, $\boldsymbol{c}$ needs to satisfy

$$c_l \geq \max\left(-h_l^{+0}, -h_l^{-0} \right) , \quad l = 0, 1, \cdots, NL-1. \quad (13.114)$$

Therefore, the following values for u_1 and u_2 are all valid:

$$u_1 = \sum_{l=0}^{NL-1} h_l^+ = \sum_{l=0}^{NL-1} h_l^{+0} + \sum_{l=0}^{NL-1} c_l , \quad (13.115)$$

$$u_2 = \sum_{l=0}^{NL-1} h_l^- = \sum_{l=0}^{NL-1} h_l^{-0} + \sum_{l=0}^{NL-1} c_l . \quad (13.116)$$

13.8 Summary

In this chapter we have investigated blind identification of acoustic multichannel systems with emphasis on adaptive implementations. The necessary and sufficient conditions for an acoustic SIMO system to be blindly identifiable using only second-order statistics were explained and the principle of blind SIMO identification presented. We illustrated how to use the cross-relations between different microphone outputs to define an error signal and its corresponding cost function. It was then shown that, by minimizing the cost function, an adaptive algorithm could be developed and the acoustic channel impulse responses would be determined up to a scale. We discussed a number of state-of-the-art adaptive blind SIMO identification algorithms, in both the time and frequency domains, with or without optimal step-size controls. Finally, we also discussed how to take advantage of the sparseness in the acoustic impulse responses to accelerate initial convergence of an adaptive blind identification algorithm for acoustic multichannel systems.

References

13.1 Y. Sato: A method of self-recovering equalization for multilevel amplitude-modulation, IEEE Trans. Commun. **COM-23**, 679–682 (1975)

13.2 D.N. Godard: Self-recovering equalization and carrier tracking in two-dimensional data communication systems, IEEE Trans. Commun. **COM-28**, 1867–1875 (1980)

13.3 J.R. Treichler, B.G. Agee: A new approach to multipath correction of constant modulus signals, IEEE Trans. Acoust. Speech **ASSP-31**, 459–472 (1983)

13.4 A. Benveniste, M. Goursat: Blind equalizers, IEEE Trans. Commun. **COM-32**, 871–883 (1984)

13.5 J.M. Mendel: Tutorial on higher-order statistics (spectra) in signal processing and system theory:

theoretical results and some applications, Proc. IEEE **79**, 278–305 (1991)
13.6 L. Tong, G. Xu, T. Kailath: A new approach to blind identification and equalization of multipath channels, Proc. 25th Asilomar Conf. on Signals, Systems, and Computers, Vol. 2 (1991) pp. 856–860
13.7 H. Liu, G. Xu, L. Tong: A deterministic approach to blind equalization, Proc. 27th Asilomar Conf. on Signals, Systems, and Computers, Vol. 1 (1993) pp. 751–755
13.8 M.I. Gürelli, C.L. Nikias: A new eigenvector-based algorithm for multichannel blind deconvolution of input clolored signals, Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., Vol. 4 (1993) pp. 448–451
13.9 L.A. Baccala, S. Roy: A new blind time-domain channel identification method based on cyclostationarity, IEEE Signal Process. Lett. **1**, 89–91 (1994)
13.10 G. Xu, H. Liu, L. Tong, T. Kailath: A least-squares approach to blind channel identification, IEEE Trans. Signal Process. **43**, 2982–2993 (1995)
13.11 E. Moulines, P. Duhamel, J.F. Cardoso, S. Mayrargue: Subspace methods for the blind identification of multichannel FIR filters, IEEE Trans. Signal Process. **43**, 516–525 (1995)
13.12 D. Slock: Blind fractionally-spaced equalization, prefect reconstruction filerbanks, and multilinear prediction, Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., Vol. 4 (1994) pp. 585–588
13.13 Y. Hua: Fast maximum likelihood for blind identification of multiple FIR channels, IEEE Trans. Signal Process. **44**, 661–672 (1996)
13.14 L. Tong, S. Perreau: Multichannel blind identification: from subspace to maximum likelihood methods, Proc. IEEE **86**, 1951–1968 (1998)
13.15 Y. Huang, J. Benesty: Adaptive multi-channel least mean square and Newton algorithms for blind channel identification, Signal Process. **82**, 1127–1138 (2002)
13.16 C. Avendano, J. Benesty, D.R. Morgan: A least squares component normalization approach to blind channel identification, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 4 (1999) pp. 1797–1800
13.17 S. Haykin: *Adaptive Filter Theory*, 4th edn. (Prentice Hall, Upper Saddle River 2002)
13.18 T.K. Moon, W.C. Stirling: *Mathematical Methods and Algorithms* (Prentice Hall, Upper Saddle River 1999)
13.19 H. Chen, X. Cao, J. Zhu: Convergence of stochastic-approximation-based algorithms for blind channel identification, IEEE Trans. Inform. Theory **48**, 1214–1225 (2002)
13.20 Y. Huang, J. Benesty, J. Chen: Optimal step size of the adaptive multichannel LMS algorithm for blind SIMO identification, IEEE Signal Process. Lett. **12**, 173–176 (2005)
13.21 M. Dentino, J. McCool, B. Widrow: Adaptive filtering in the frequency domain, Proc. IEEE **66**, 1658–1659 (1978)
13.22 A.V. Oppenheim, R.W. Schafer: *Discrete-Time Signal Processing* (Prentice Hall, Englewood Cliffs 1989)
13.23 D.H. Brandwood: A complex gradient operator and its application in adaptive array theory, Proc. IEE **130**, 11–16 (1983), Pts. F and H
13.24 Y. Huang, J. Benesty: A class of frequency-domain adaptive approaches to blind multichannel identification, IEEE Trans. Signal Process. **51**, 11–24 (2003)
13.25 J. Benesty, D. Morgan: Frequency-domain adaptive filtering revisited, generalization to the multi-channel case, and application to acoustic echo cancellation, Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., Vol. 2 (2000) pp. 789–792
13.26 D.L. Duttweiler: Proportionate normalized least-mean-square adaptation in echo cancelers, IEEE Trans. Speech Audio Process. **8**, 508–518 (2000)
13.27 J. Benesty, Y. Huang: The LMS, PNLMS, and exponentiated gradient algorithms, Proc. EUSIPCO (2004) pp. 721–724
13.28 J. Kivinen, M.K. Warmuth: Exponentiated gradient versus gradient descent for linear predictors, Inform. Comput. **132**, 1–64 (1997)
13.29 S.I. Hill, R.C. Williamson: Convergence of exponentiated gradient algorithms, IEEE Trans. Signal Process. **49**, 1208–1215 (2001)
13.30 J. Benesty, Y. Huang, J. Chen: An exponentiated gradient adaptive algorithm for blind identification of sparse SIMO systems, Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., Vol. 2 (2004) pp. 829–832

11. Formant Estimation and Tracking

D. O'Shaughnessy

This chapter deals with the estimation and tracking of the movements of the spectral resonances of human vocal tracts, also known as formants. The representation or modeling of speech in terms of formants is useful in several areas of speech processing: coding, recognition, synthesis, and enhancement, as formants efficiently describe essential aspects of speech using a very limited set of parameters. However, estimating formants is more difficult than simply searching for peaks in an amplitude spectrum, as the spectral peaks of vocal-tract output depend upon a variety for factors in complicated ways: vocal-tract shape, excitation, and periodicity. We describe in detail the formal task of formant tracking, and explore its successes and difficulties, as well as giving reasons for the various approaches.

11.1 Historical

In this chapter, we deal with a focused problem of speech analysis – trying to identify some very specific aspects of speech that have been found to be of great use in a wide variety of speech applications. These parameters of speech are called formants. They are generally viewed to be the resonances of the vocal tract (VT), which often appear in spectral displays (such as spectrograms) as regions of high energy, slowly varying in time as the vocal tract moves (Fig. 11.1).

Formants are useful in the coding, recognition, synthesis, and enhancement of speech, as they efficiently describe essential aspects of speech using a very limited set of parameters. For coding, if speech can be reduced to formant parameters to represent the VT shape (and a few other parameters to represent VT excitation), then very efficient coding is possible [e.g., 2.4 kbps linear predictive coding (LPC)]. Standard coders (e.g., in cellphone technology) use LPC with 10 or so coefficients to characterize the VT; it is feasible instead to represent the VT with even fewer parameters if formants are properly employed.

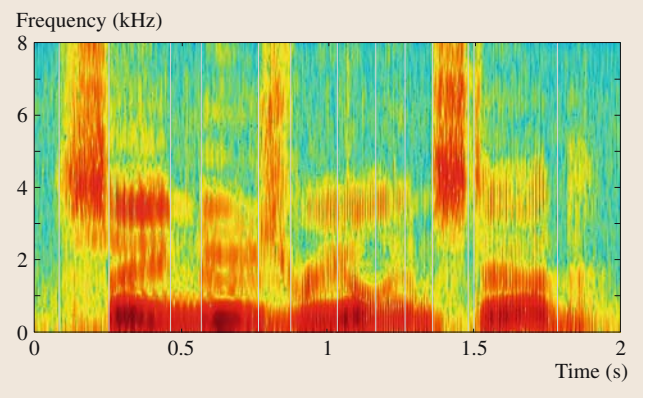


Fig. 11.1 Wideband spectrogram of an adult male speaker saying 'Say newsreel instead'. *Light vertical lines* (added manually) denote phoneme boundaries

In the 1970s, formants were the primary focus of automatic speech recognizers (ASRs), as it is necessary in ASR (as in coding) to greatly reduce the information present in a speech signal (e.g., 64 kbps for toll-quality speech in the basic telephone network), without sacrificing useful information about VT shape [11.1]. As many studies of human speech production and perception have identified formants as prime candidates to represent the VT spectrum efficiently, they were popular parameters to try to estimate from speech signals. It was found, however, that tracking formants reliably was not an easy task. Since the mid-1980s, ASR has primarily relied on the mel-frequency cepstral coefficients (MFCCs) instead of formant-based parameters to represent VT information. The advantage of the MFCC approach has been an automatic way to reduce the amount of information in a Fourier transform (FT) of a frame of speech (which is always assumed to reasonably capture the essential information about VT shape at any specific point in time; a frame is a short section of speech, e.g., 20 ms) to a small set of parameters, e.g., 10–16. The data reduction factor is about the same as for LPC, except that the MFCC is able to utilize some auditory factors in warping frequency scales to model the human ear better than LPC can (i. e., a mel or bark scale: linear up to 1 kHz, and logarithmic thereafter). Nonetheless, there still remains interest in formants for ASR purposes, as MFCC and LPC tend to suffer significantly when increasing amounts of noise are present in the received speech signal. Both MFCC and LPC take global approaches to speech analysis, which makes it difficult to separate noise from speech in corrupted signals.

The MFCC and LPC have been so popular for speech recognition and coding, respectively, because they are parameters that are obtained by simple mathematical rules (algorithmic transformations not subject to discrete, and hence nonlinear, decisions). Such transformations reduce the size of speech representations. A Fourier display does little data reduction as it occupies about as much data space as the untransformed speech. The MFCC and LPC are more compressed, but still leave room for further compression, as is possible with formants. *Features* such as formants must be estimated using error-prone methods, as they can only be obtained by applying (possibly faulty) decisions in the data reduction process. They achieve greater data reduction, at the cost of making errors. Given the widespread use of MFCC for ASR, some formant trackers employ the MFCC as the spectral input to their algorithms [11.2, 3], even though the MFCC tend to smooth spectra in ways that may obscure complicated formant structure. (Because ASRs using MFCCs rarely track formants, any formant tracking errors that might result from such an approach would not be pertinent for ASR, but it is nonetheless useful to see how well one can track formants using such a popular spectral estimation method.)

If instead we return to an original (possibly noisy) spectrum and properly examine the short-time Fourier transform (STFT) to find formants amid any interfering noise, there is still potential to carry out better ASR with formants [11.4]. Recently, formants have seen increasing use in text-to-speech (TTS) synthesis applications [11.5], as the trend has been to employ very large databases of small speech units (extracted from the speech of a single professional training speaker). In TTS, given a text to pronounce, a phonetic sequence is automatically determined, allowing access to these units. A major issue in recent research has been the efficient determination of which units to use, which requires searching a large space of alternatives (typically, through a Viterbi search) making many cost estimations about the spectral similarity between units. If these units are characterized via formants, these costs are often easier to compute than if other parameters are used.

Finally, aids for people with speech difficulties are often designed around the essential aspects or features of speech such as formants. In some speech aids, information additional to the normal speech input may be available to assist in the formant tracking task. Attaching a laryngeal sensor to a speaker's throat can provide detailed information about the VT excitation in a relatively unobtrusive fashion, and appears to assist in some formant tracking methods [11.6].

Certainly, a visual display capturing the mouth of the speaker corresponding to the input speech can help both ASR and formant tracking, as such imagery yields important information about (at least) mouth opening. As most speech applications do not have access to imagery, but only to the speech itself, we will not assume image assistance in this section. There are several formant trackers in commercial applications; a common one freeware package is Wavesurfer [11.7].

11.2 Vocal Tract Resonances

While formants are commonly viewed as peaks in the speech spectrum, we must be more rigorous in defining them, as spectral displays of speech vary greatly in the number and types of peaks seen (Fig. 11.2). Formants are usually understood to be broad spectral peaks in an STFT of speech, corresponding to the underlying vocal tract resonances (VTRs). Such basic resonances can often be calculated from VT area functions, if available, e.g., via X-rays or electromyography. Few speech applications have access to such data, however, and thus resonance estimations must usually be based on analysis of the speech coming from the mouth. However, such resonances are only present in output speech energy to the extent that the VT system is excited with sufficient energy at those frequencies. We must recall that speech output is the convolution of the VT excitation waveform and the impulse response of the VT (or, equivalently, the speech spectrum is the product of these two spectral representations), and thus the nature of the VT excitation is a major factor in whether VTRs form visible peaks in the output speech STFT.

As we are primarily interested in formant estimation during strong sonorants, we first look at VT excitation for vowels, where the excitation consists of glottal puffs of air modulated by the vocal folds with a low-pass nature (about −12 dB/octave fall-off). As a result, the first formant (F_1) normally has the highest intensity, and the amplitude decreases at about −6 dB/octave for formants at higher frequencies (−6 is the net result, after including the +6 dB/octave radiation effect at the lips). Spectrograms often show clear bands of formant energy for the first 3–5 formants in vowels, but this situation varies greatly depending on the spectrum of each vowel [e.g., /u/ has a greater fall-off, with little energy above F_2 (second formant), as its F_1 and F_2 are quite low in frequency, whereas /i/, with a much higher frequency F_2, has much more energy in the F_3–F_4 region]. The style of speech also has effects: breathy voice has a greater rate of fall-off, and hence weaker formants at high frequencies, than a shouting voice. Few speech applications have shown interest in the estimation of formants above F_4, as higher formants are quite weak (especially relative to any background noise) and have much less perceptual relevance than F_1–F_4. Indeed, estimations of F_3 and F_4 are often difficult owing to the low energy in their frequency ranges (e.g., for back vowels such as /u/).

Issues of weak formants are not limited to high frequencies. Nonvowel phonemes, such as nasals and obstruents, partly owing to the presence of zeros in their spectra, have varied spectral displays in terms of which resonances are visible. In the case of obstruents, a major additional factor is the different nature and location of the noise excitation, which is much higher in the VT, hence exciting only higher resonances, as will be

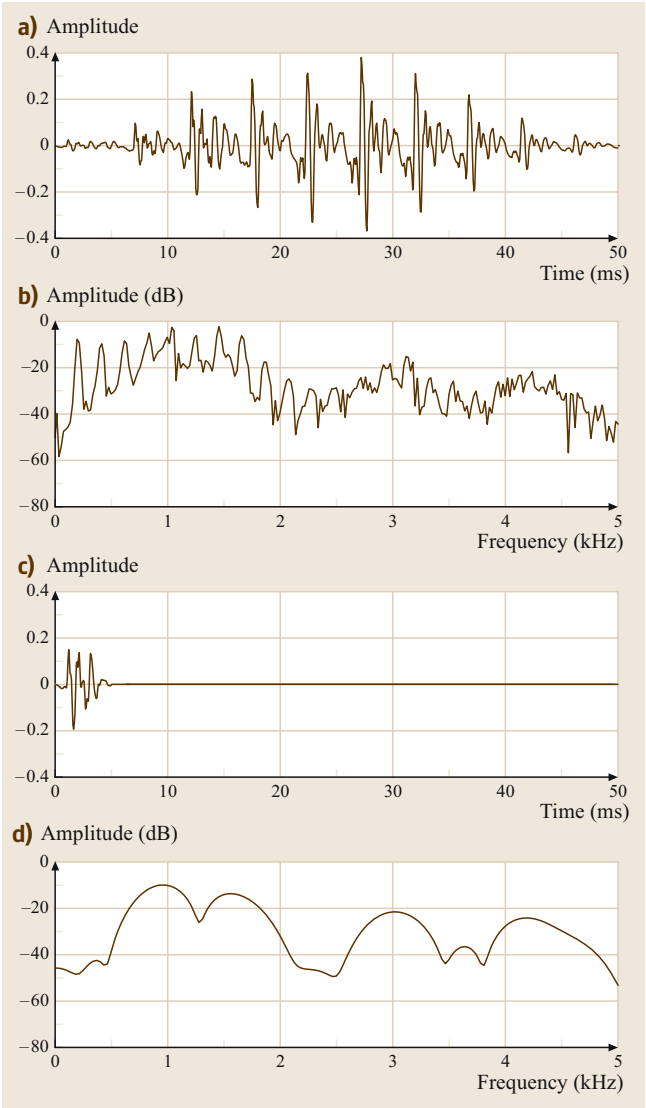


Fig. 11.2a–d Time signals and spectra of a vowel. **(a)** Speech signal weighted by a 50 ms Hamming window, **(b)** the corresponding spectrum, **(c)** speech signal weighted by a 5 ms Hamming window, and **(d)** the corresponding spectrum

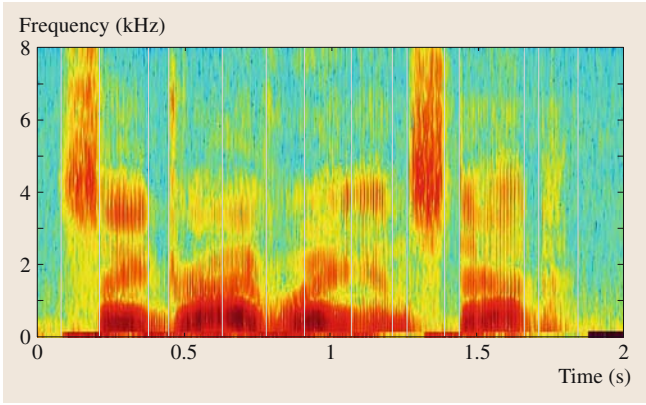


Fig. 11.3 Wideband spectrogram of the same speaker saying 'Say driveway instead'

discussed later. Thus, the varying display of bands of energy that we wish to associate with VTRs is a major factor in the difficulty of formant tracking, and not just for a minority of speech sounds.

Part of the difficulty of formant estimation and tracking has been not properly exploiting much of what is known about human speech production and perception. This failing has occurred too often in the speech research field (e.g., for many years, ASR ignored the fact that speech had an underlying message component; it was only in the 1980s that language models were commonly used for ASR). A major difficulty of tracking formants is their tendency in spectral displays to weaken, disappear, and/or merge (Figs. 11.1–11.3), as the VT undergoes changes in shape or as the VT excitation changes. As speech is inherently dynamic (e.g., 12 phonemes/second on average), the ability to track temporal changes in any parametric representation (e.g., formants) is essential. While such changes (e.g., abrupt appearance or disappearance of formants, as well as formants approaching each other so closely as to appear as one band of energy in time displays) are readily apparent, the underlying resonances of the VT are much more well behaved, albeit harder to observe. VTRs derive directly from VT shape, and are the poles of the VT transfer function. As the VT moves smoothly from one position to another, the VTRs smoothly change values [11.8]. Even when occlusions are made or released (as in stop consonants), the positions of the VTRs change smoothly. As such, VTRs would be much easier to track than formants (whose appearance on spectrograms is more varied). However, lacking visual displays of the VT (e.g., a camera focused on the speaker's mouth, or a radiographic image of the inside of the mouth), which is normally the case in almost all speech applications, we must rely on formant rather than VTR tracking. We can, nonetheless, exploit presumed knowledge about VTRs when estimating formants.

11.3 Speech Production

To better understand the foundation for formants, let us examine some aspects of human speech production. In any estimation task, it is often useful to examine important properties of the data source, rather than attack a problem with few assumptions. Too often in the past, research on formant estimation simply viewed STFTs, looked for bands of energy, and applied basic peak-picking approaches (with some constraints on continuity), without further regard for the nature or origin of the speech signal. It is true that the essential information about formants (for most speech applications) is readily available in such displays, but such an approach is like doing ASR without a language model. An essential step of any pattern recognition task, whether complete ASR or simpler formant tracking, is to reduce the input data while minimizing loss of useful information; this *preprocessing* step not only makes the process more efficient (as subsequent recognition steps treat smaller amounts of data), but also allows more-accurate estimation decisions, by focussing more closely on aspects of the data that are crucial (for the task at hand). In many cases of pattern recognition, an underlying source undergoes several (often nonlinear) mappings before a data acquisition device can capture a readily observable signal. A better understanding of these mappings can aid the parameter estimation process. In our case, the basic underlying source is a message in a speaker's brain. ASR attempts to recover that message (e.g., a text), given the observed speech signal. More specifically here we are concerned with identifying certain intermediate parameters – the formants – in this process. In any event, better knowledge about where formants originate is useful.

In human speech production, the lungs (as a pressure source) push air past the vocal folds, which often modulate the airflow to create the excitation sound for

the VT, which in turn acts as a filter to amplify certain frequencies while attenuating others. The VT is the most important component in speech production, as it controls the formants and its variation leads directly to the perception (by listeners) of the individual phonemes (sounds) of speech. A tubular passageway composed of muscular and bony tissues, the VT, via its shape, modifies the spectral distribution of energy in glottal sound waves. Different sounds of speech are primarily distinguished by their periodicity (voiced or unvoiced), spectral shape (which frequencies have the most energy), and duration. The vocal folds specify the voicing feature, but the major partitioning of speech into sounds is accomplished by the VT via spectral filtering. The VT is often modeled as an acoustic tube with resonances called formants, but it can have antiresonances as well. In z-transform terminology, each resonance corresponds to a complex conjugate pair of poles, whose angular position corresponds to the center frequency of the resonance (usually, we refer to this as the formant value) and whose radial position corresponds to the resonance bandwidth. (In formant tracking, the amplitudes of the resonance peaks are usually of considerably less interest than the center frequencies and bandwidths, as amplitudes are greatly influenced by the spectral tilt of the glottal excitation, as well as by the proximity of nearby formants; listeners seem to take account of this by utilizing rather little relative peak amplitude information in speech perception.) The formants are often abbreviated as F_i, e.g., F_1 is the formant with the lowest frequency. (F_0 is associated with the fundamental frequency of the vocal folds during voiced speech, and is not a resonance. F_0 is quite visible in narrowband spectrograms, as speech consists of F_0 and its many harmonics, modulated by the VT transfer function, i. e., the spectral envelope.)

Figure 11.2 shows displays of the amplitude of an STFT for a typical vowel. If, as is usual, we use a time window (i. e., the length of the STFT) of about 20–25 ms, perhaps shaped by a Hamming window to reduce edge effects [11.1], the display will show a large amount of detail, corresponding to many aspects of the windowed speech signal. (Typical FFT lengths are 256, 512, or 1024 samples, which correspond to common frame durations at sampling rates around 16 kHz; as a result, sample points in the STFT are about 30 Hz apart, which means that each harmonic has a few values, and the STFT is highly variable in frequency.) Most notable are the harmonics, peaks spaced every F_0 Hz (Fig. 11.2b), which correspond to multiples of the vocal fold vibration rate. (These equally spaced peaks have finite width, corresponding to the inverse of the window duration; if one were to theoretically reiterate a vowel's pitch period indefinitely, we would see an actual line spectrum.) Superimposed on this base periodic set of spectral lines is the VT transfer function or spectral envelope, as the STFT is the product of the excitation spectrum (F_0 lines) and the VT filter response. It is this spectral envelope, and more particularly its formant peaks, that interest us the most in this Section. Note, however, that the formants are not immediately accessible in the FT, as the spectral envelope is essentially only specified at multiples of F_0. Typically, one may smooth the STFT to render the formants more evident.

Another way to suppress the spectral effects of the (interfering) harmonics is to use an STFT with a shorter time window (i. e., one shorter than the pitch period) (Fig. 11.2d). A disadvantage of this approach is that the shorter window needs to be placed synchronously with the pitch period for optimal formant estimation, and any corrupting noise in the speech signal has greater negative effects on formant estimation, as a much shorter set of speech data is used (owing to the shorter window).

Antiresonances correspond to zeros in the output speech, and occur owing to aspects of glottal excitation or to the existence of side-branches in the airflow path in the VT. This section examines formants, which correspond to the resonances. As such, we will not explicitly try to track the zeros, which are considerably more difficult to locate reliably (very recently, particle filtering has been used to track the poles and zeros of the VT [11.9]). Luckily, zeros are much less relevant for most speech applications.

Figures 11.1 and 11.3 show typical wideband spectrograms. Many speech applications make reference to such spectrograms, although few actually include these three-dimensional displays explicitly in their speech analysis (e.g., LPC and the MFCC are much more commonly used in coding and in ASR, respectively). They clearly show the formant bands of energy (where darkness indicates intensity) as a function of both time and frequency. In wideband spectrograms, use of a short time window yields much better time resolution (e.g., about 3 ms, making the vocal fold closure excitations readily visible as increases in energy every pitch period). As a result, frequency is typically smoothed automatically over a range of 300 Hz (roughly the inverse of 3 ms, and chosen to exceed most people's F_0, and hence include two or more harmonics within the window's low-pass filter range of smoothing). In such a dis-

play, the time signal is covered roughly continuously, thus obviating the need for pitch-synchronous analysis; a frame-based analysis method, as in ASR applications, would, on the other hand, need to address that issue.

11.4 Acoustics of the Vocal Tract

Speech is produced when air passes through the VT, which can be modeled as an acoustic tube of variable cross-sectional area $A(x, t)$, approximately closed at the glottal end and roughly open at the lips [A varies in space ($x = 0$ at the glottis and $x = L$ at the lips; L will be assumed to be 17 cm) and in time t]. VT length varies greatly among speakers (about 13 cm for women, and less for children); we choose 17 cm as the nominal value here as this is an average value for men, and yields simple values for the average positions of the formants: 500 Hz, 1500 Hz, 2500 Hz, 3500 Hz, ... etc. (as we will see later). Glottal area is small relative to typical A values, although the glottal end of the VT is truly closed only during glottal stops and during the closed phases of voicing. Lip rounding or closure often narrows the acoustic tube at the lips. A tube (e.g., the VT) closed at one end and open at the other resembles an organ pipe and is called a quarter-wavelength resonator, as the frequencies at which the tube resonates are those where sound waves traveling up and down the tube reflect and coincide at the ends of the tube. VTRs can be heuristically computed using only the boundary conditions of the VT, along with the phase relationship between the pressure and volume velocity in the traveling sound waves. Formant frequencies match the boundary conditions for pressure P (relative to atmospheric pressure) and volume velocity U: a closed end of the tube makes $U = 0$, whereas $P \approx 0$ at an open end. P is 90° out of phase with U, owing to the inductance and capacitance of the VT. Resonances occur at frequencies F_i, $i = 1, 2, 3, \ldots$, where $|U|$ is maximum at the open end of the VT and $|P|$ is maximum at the closed end. Such frequencies have wavelengths where the VT length l is an odd multiple of a quarter-wavelength; hence, at 500 Hz, 1500 Hz, 2500 Hz, ... etc..

A uniform VT is only a good model for a schwa vowel /ə/. With other sounds, $A(x)$ is a complicated function of space along the VT, and as a result, the formants move to other frequencies. Normally, the deviations are within a range of a few (or several) hundred Hz. Thus, for a 17 cm VT, F_1 is usually in the range 300–800 Hz; F_1 tends to be low when the VT is relatively closed (e.g., for most consonants, and for vowels with a raised tongue – high vowels), and high for low vowels. F_2 is usually in the range 700–2200 Hz; F_2 tends to be high when the tongue is relatively forward, and low when the tongue is more to the rear. In general, the range of F_2 is greater than for the other formants. F_3 is usually in the range 1800–2800 Hz; F_3 tends to be high when the tongue is relatively forward and high, and low when the tongue is retroflexed (as in /r/). Higher formants are usually progressively weaker in intensity, and less relevant for most applications. (Synthesizers that use formants often fix F_4 near 3500 Hz and F_5 near 4500 Hz; variations in these numbers appear to have little useful perceptual effect.) Thus, formant estimators focus on tracking F_1–F_3.

Almost all languages employ the *cardinal* vowels of /i/, /a/, /u/, and these three often represent VT shapes that are far from a neutral schwa (uniform VT) shape, i. e., other vowels (if present for a given language) have more intermediate shapes, and therefore formant frequencies that are closer to the neutral values (500, 1500, 2500 Hz). Tracking formants is typically more difficult for more extreme VT shapes, as some formants often appear to merge in such cases. (Examples are seen in Figs. 11.1 and 11.3, where formants change intensity suddenly and make apparent jumps, usually at phoneme boundaries, but certainly not at all phoneme boundaries.) Thus, let us examine further how VT shape relates to such vowels. The vowel /a/ can be roughly modeled by a (lower) narrow tube (representing the pharynx) opening relatively abruptly into a wide (upper) tube (the oral cavity). Assuming a 17 cm VT, for simplicity suppose that each tube has a length of 8.5 cm; then, each tube would produce the same set of resonances, at odd multiples of 1 kHz (1000, 3000, 5000 Hz, ...). Each tube is a quarter-wavelength resonator, since its back end is relatively closed and its front end is relatively open (i. e., the two tubes are half-versions of the full schwa VT, and thus have formants at twice the original values). As with all 17 cm models of the VT, this two-tube model has the same number (one) of formants per kHz, on average, but each one is moved by 500 Hz.

11.4.1 Two-Tube Models for Vowels

At any boundary between sections of the VT, whenever the change in areas is sufficiently abrupt (as with /a/, near the velum), the acoustic coupling between cavities

is small and the interaction between cavity resonances will be slight; each section then controls its own number of the overall set of formants. Due to some acoustic coupling, formants never approach each other by less than about 200 Hz; thus, e.g., F_1 and F_2 for /a/ are not both at 1000 Hz, but rather near these values: $F_1 = 900$ Hz, $F_2 = 1100$ Hz, $F_3 = 2900$ Hz, and $F_4 = 3100$ Hz. The reverse holds for /i/: a wide pharyngeal tube narrowing abruptly into the oral cavity tube. Theory would have $F_1 = 100$ Hz, $F_2 = 1900$ Hz, $F_3 = 2100$ Hz, and $F_4 = 3900$ Hz, but in practice, F_1 is closer to 280 Hz, and F_4 often approaches F_3 (making a group of 2–3 formants around 2 kHz). Actual observed values for real versions of /a/ and /i/ will show deviations from these model numbers due to modeling inaccuracies; nonetheless, these simple models give reasonably accurate results and are easy to interpret physically.

The third cardinal vowel /u/ has a more-complicated analysis: anytime that the lips are rounded (as in /u/), all formants lower in frequency. This can be roughly seen by considering a VT mostly closed at both ends; the VT then becomes a half-wavelength resonator, as its boundary conditions are the opposite of the normal open-mouth model (e.g., the volume velocity is minimized at both (closed) ends of the VT, whereas it is maximized at the mouth in the open-mouth version). The theoretical locations for such a VT model are $F_1 = 0$, $F_2 = 1000$ Hz, $F_3 = 2000$ Hz, etc. Again, in practice, F_1 rarely goes so low, and approaches 200 Hz as the lips are almost closed. Examining such an analysis, one may be tempted to conclude that formants do not deviate more than about 400 Hz from the nominal neutral values of 500, 1500, 2500 Hz. Empirical evidence, however, shows that F_2, in particular, has a rather wider range (and F_3 goes as low as 1900 Hz). Again, all values given here assume a typical man's VT; for people of other sizes, one generally scales frequency values by the average length of the VT, relative to 17 cm; as VTs are not linear versions of each other, there may be significant deviations from such simple modeling.

11.4.2 Three-Tube Models for Nasals and Fricatives

We cannot easily extend the simple results of the two-tube VT model to more-complicated VT shapes for other phonemes. Nonetheless, some approximate extensions are feasible, and help understanding of the task of formant estimation. Nasal consonants require analysis of a three-tube VT model, as the nasal cavity is involved when the velum is lowered for such consonants. The entire system can be modeled as three tubes (pharyngeal, nasal, and oral) joined at one point (the velum), with acoustic circuits for the three in parallel. The poles of such a model are specified by [11.1]: $1/Z_\mathrm{p} + 1/Z_\mathrm{m} + 1/Z_\mathrm{n} = 0$, where $Z_\mathrm{p} = -\mathrm{i}Z_{0\mathrm{p}}\cot(\beta l_\mathrm{p})$, $Z_\mathrm{m} = -\mathrm{i}Z_{0\mathrm{m}}\cot(\beta l_\mathrm{m})$, and $Z_\mathrm{n} = \mathrm{i}Z_{0\mathrm{n}}\tan(\beta l_\mathrm{n})$. Z_p, Z_m, and Z_n refer to the acoustic impedance seen in the velar region of the VT, in the direction of the pharynx, nasal passages, and mouth, respectively, while $Z_{0\mathrm{p}}$, $Z_{0\mathrm{m}}$, and $Z_{0\mathrm{n}}$ are the characteristic impedances of these respective cavities. The mouth and pharyngeal tubes have closed acoustic terminations, while the nasal tube is open (at the nostrils). Dimensional similarity of the pharyngeal and nasal tubes allows a simplification: if $l_\mathrm{p} = l_\mathrm{n} = 10.5$ cm, each of $1/Z_\mathrm{p}$ and $1/Z_\mathrm{n}$ has periods of about 1.6 kHz and the function $1/Z_\mathrm{p} + 1/Z_\mathrm{n}$ has infinite values about every 800 Hz.

The mouth tube for nasal consonants is often significantly shorter than the other tubes (e.g., 3–7 cm). As a result, nasal consonants are characterized by: (a) formants every 800 Hz (due to the longer pharynx+nasal tube than the normal pharynx+mouth tube), (b) wider formant bandwidths, and (c) zeros in the spectrum. When airflow from the lungs reaches the velum junction, it divides according to the impedances of the mouth and nasal tubes. Spectral zeros occur at frequencies where $Z_\mathrm{m} = 0$, which results in no airflow into the nasal tube and thus no nasal speech output. Solving $Z_\mathrm{m} = -\mathrm{i}Z_{0\mathrm{m}}\cot(2\pi F_i l_\mathrm{m}/c) = 0$ for F_i yields zeros at odd multiples of $c/4l_\mathrm{m}$. The mouth tube for /m/ is about 7 cm, which gives zeros at 1.2, 3.6, 6.0 kHz, ... Shorter tubes for /n/ and /ŋ/, about 5 and 3 cm, respectively, mean fewer zeros below 5 kHz: only one each at 1.7 and 2.8 kHz, respectively. Besides the poles due to the pharynx and nasal cavities, which occur every 800 Hz, nasal spectra have *pole–zero pairs* due to the mouth cavity; i. e., each zero is close in frequency to a mouth cavity pole. In spectra, nasal consonants appear as sounds with relatively steady formants, weaker than for vowels and with less coarticulation, as movements within the VT are muffled by oral tract closure.

Unlike the sonorants, all obstruents (except the glottal /h/) have an excitation source in the (upper) oral cavity. Air passing through a narrow constriction there creates frication noise just in front of the opening, which excites the remainder of the oral cavity in front of the constriction. This is usually modeled as random (Gaussian) noise with an approximately flat spectrum. For voiced obstruents (e.g., /v/ and /z/), the noise is modulated to be pulse-like, as the vocal folds vibrate; in spectrograms, this adds a *voice bar* at very low fre-

quencies (0–150 Hz). This bar is weak enough not to be confused with formants, as well as being outside the range of F_1.

11.4.3 Obstruents

As noted above, formants are dynamic, varying greatly in time and intensity. They often come close together, and at other times fade in or out. Such changes are due to VT movements, and to the nature of VT excitation. Voiced excitation is glottal and usually relatively strong, but its intensity decreases with increasing frequency, owing to the low-pass nature of glottal puffs of air. As a result, a voiced speech spectrum falls off with frequency at about -6 dB/octave, which leads to weaker high-frequency formants. For most obstruent sounds (i. e., stops and fricatives) the VT excitation occurs much higher in the VT than at the glottis (except for the fricative /h/), and as a result, a much shorter VT is excited. This (often much) shorter VT leads to much higher resonances appearing in speech spectra, or more precisely, to the lower-frequency resonances being canceled by antiresonances of the back cavity of the VT. Many researchers do not refer to details of obstruent spectra as *formants* (i. e., they reserve the term formants for spectra of sonorant phonemes); to describe obstruents, they instead note general details such as the approximate cutoff frequency below which little energy occurs. The justification for this is that listeners pay much less attention to detailed aspects of the spectra in obstruents, and speakers appear to exercise less control over the positioning of resonances there. Hence, we focus our attention on formant estimation for sonorants.

11.4.4 Coarticulation

The motion of the VT during speech causes very dynamic patterns in formants, i. e., we often see rapid movements of formants. This phenomenon is called coarticulation, because the articulatory configurations of neighboring phonemes affect the articulation of each phoneme. Even though text is written with discrete letters, and phoneticians note that speech consists of a sequence of individual phonemes, the actual speech signal is without obvious phoneme boundaries, owing to coarticulation. Speakers smoothly move their VT from positions appropriate for each phoneme in turn, spending as much time on each phoneme as is deemed appropriate. As a result, it is often hard to segment speech into discrete units for analysis, as would be very useful in ASR. In some cases, such a division is easy, as when excitation changes abruptly with the start or end of vocal fold vibration (i. e., a voiced–unvoiced transition), or when the oral portion of the VT closes or opens (e.g., lip closure). More often, the transition between phonemes is more subtle, as the tongue and lips move between positions appropriate for successive phonemes.

Each phoneme has a nominal or *target* VT shape that a speaker would more or less assume if the sound were to be produced in isolation. In context, however, the speaker thinks ahead, and is continually moving various parts of the VT to accomplish the dynamic sequence of phonemes. Thus, coarticulation effects can extend over several phonemes; e.g., in the word 'strew' (/stru/, phonetically), the lips round in anticipation of the /u/ during the earlier /s/.

In many applications, it is useful to divide speech into segments of linguistic relevance, e.g., words, syllables, or phonemes. For example, for speech synthesis-by-rule (TTS), continuously spoken speech from a training speaker must be segmented into such units for storage (for later concatenation, as needed for a specified input text). As manual segmentation is tedious, forced alignment is often imposed on such training speech, given the assumed corresponding text. As a result, we have applications where formant tracking is simplified by having a priori knowledge of what phoneme sequence was actually spoken [11.10]. Such studies report quite high accuracy of forced alignment to phonemic boundaries in speech of a known text, to within about 40 ms [11.11]. In more-general ASR applications, of course, we do not know beforehand what was said, and formant estimation must rely on general principles.

During slow speech, the VT shape and type of excitation may not alter for periods of up to 200 ms. In these sections of speech, formant tracking is usually much easier, as the formants vary little in either position or amplitude. Most of the time, however, the VT changes more rapidly, as phonemes last, on average, about 80 ms. Coarticulation and changing F_0 can render each pitch period different from its neighbor. Nonetheless, a basic assumption of speech analysis is that the signal properties change relatively slowly with time. This allows examination of a short time window of speech (e.g., multiplying the speech signal by a Hamming window) to extract parameters presumed to remain fixed for the duration of the window. Most techniques thus yield parameters averaged over the course of the time window. To model dynamic parameters, we divide the signal into successive windows (ana-

lysis frames). Slowly changing formants in long vowels could allow windows as large as 100 ms without obscuring the desired parameters via averaging, but rapid events (e.g., stop releases) need short windows of about 5–10 ms to avoid averaging spectral transitions with the steadier spectra of adjacent sounds.

11.5 Short-Time Speech Analysis

A basic tool for spectral analysis is the spectrogram, which converts a two-dimensional speech waveform (amplitude versus time) into a three-dimensional pattern (amplitude/frequency/time). With time and frequency on the horizontal and vertical axes, respectively, amplitude is noted by the darkness of the display (Fig. 11.1). Peaks in the spectrum appear as dark horizontal bands. The center frequencies of these bands are generally considered to be the formant frequencies (subject to the discussion below of how to handle merged formants, i. e., single, wider bands that display two or more resonances that have come close for certain periods of time; e.g., in the middle of Fig. 11.3, F_1 and F_2 combine at very low frequencies in /w/). Voiced sounds cause vertical marks in the spectrogram due to an increase in the speech amplitude each time the vocal folds close. The noise in unvoiced sounds causes rectangular dark patterns, randomly punctuated with light spots due to instantaneous variations in energy. Spectrograms portray only spectral amplitude, ignoring phase information, on the assumption that phase is less important for most speech applications. We will thus ignore phase in our discussion of formants.

In the spectrogram, the amplitude of the STFT $|S_n(e^{i\omega})|$ is plotted with time n on the horizontal axis, frequency ω (from 0 to π) on the vertical axis (i. e., 0 to $F_s/2$ in Hz, F_s being the sampling frequency), and with magnitude indicated as darkness, typically on a logarithmic scale (e.g., decibels). Two different display styles are typical: wideband and narrowband, with wideband displays used mostly for formant tracking.

Wideband spectrograms display individual pitch periods as vertical striations corresponding to the large speech amplitude each time the vocal cords close. Voicing is readily seen in the presence of these periodically spaced striations. Fine time resolution here permits accurate temporal location of spectral changes corresponding to VT movements. A wide filter bandwidth smooths the harmonic amplitudes under each formant across a range of (typically) 300 Hz, displaying a band of darkness (of width proportional to the formant's bandwidth) for each formant. The center of each band is a good estimate of formant frequency. Formant detectors generally prefer spectral representations that smooth the fine structure of the harmonics while preserving formant structure. Traditional wideband spectrograms use a window of about 3 ms, which corresponds to a bandwidth of 300 Hz and smooths harmonic structure (unless $F_0 > 300$ Hz, which occurs with children's voices). Narrowband spectrograms, on the other hand, generally use a window with a bandwidth of approximately 45 Hz and thus a duration of about 20 ms, which allows resolution of individual harmonics (since $F_0 > 45$ Hz) but smooths speech in time over a few pitch periods.

11.5.1 Vowels

Vowels are voiced and have the greatest intensity of all phonemes. They normally range in duration from 50 to 400 ms [11.1]. Vowel energy is mostly concentrated below 1 kHz and falls off at about −6 dB/oct with frequency. Spectral displays thus often use pre-emphasis to boost higher frequencies to facilitate formant tracking; e.g., LPC treats all frequencies the same, so LPC analysis without pre-emphasis would tend to have poorer spectral estimates at higher frequencies. Other formant methods that rely on peak-picking of spectra would likely have similar difficulties without pre-emphasis. As such boosting also raises the level of any background noise, it should be noted that formant estimation for high frequencies where noise dominates will necessarily be less accurate. Because of the −6 dB/oct fall-off, few formants above F_4 are reliable in many formant estimation methods. Pre-emphasizing the speech is usually done by differencing in discrete time:

$$y(n) = s(n) - as(n-1)\,,$$

where a is typically 0.9–1.0. While this may greatly attenuate frequencies below 200 Hz, such low frequencies are rarely of interest in most speech applications. Vowels are distinguished primarily by the locations of their first three formant frequencies (F_1, F_2, and F_3).

11.5.2 Nasals

In nasal consonants, F_1 near 250 Hz dominates the spectrum, F_2 is usually very weak, and F_3 near 2200 Hz

has the second-highest formant peak. A spectral zero, whose frequency is inversely proportional to the length of the oral cavity behind the constriction, occurs near 1 kHz for /m/, near 2 kHz for /n/, and above 3 kHz for the velar nasal. Spectral jumps in both formant amplitudes and frequencies coincide with the occlusion and opening of the oral tract for nasals. These abrupt changes cause difficulties for the continuity constraints for formant trackers, as the usual trend toward smooth formant movements is invalid at nasal boundaries. VTRs change abruptly when the oral cavity opens or closes. The lowering of the velum is not the principal factor in the spectral change here; it often lowers during a vowel preceding a nasal consonant, which causes nasalization of the vowel, widening the formants and introducing zeros into the spectrum. Vowel nasalization primarily affects spectra in the F_1 region.

11.5.3 Fricatives and Stops

As obstruents, fricatives and stops are very different from sonorants: aperiodic, much less intense, and often with most energy at high frequencies. Obstruents may be either voiced or unvoiced. Unvoiced fricatives have a high-pass spectrum, with a cutoff frequency approximately inversely proportional to the length of the front cavity of the VT. Thus the palatal fricatives are most intense, with energy above about 2.5 kHz; they have a large front cavity. The alveolar fricatives (e.g., /s/) lack significant energy below about 3.2 kHz and are thus less intense. The labial and dental fricatives are very weak, with little energy below 8 kHz, due to a very small front cavity. The glottal fricative /h/, despite exciting the full VT, also has relatively low intensity as its noise source at the glottis (effectively a whisper) is usually weaker than noise from oral tract constrictions.

It is not obvious how to handle formant tracking for obstruents. Traditionally, formants are well defined only for sonorant sounds, where the general rules of strong resonances, spaced roughly every 1000 Hz, apply. Except for /h/, obstruents have little energy in the low-frequency range of 0–2 kHz, where strong F_1 and F_2 (for sonorants) have most energy. Depending on the length of the VT in front of the constriction noise source, there may be little energy in most of the useful auditory range. As noted above, VTRs are always present, no matter where the excitation is; however, low-frequency VTRs are not excited in most obstruents, and thus are not accessible in speech analysis.

In a transition from a sonorant to an obstruent (the observations that follow here also apply, in reverse, for a transition from an obstruent to a sonorant), the visible formants usually show movements from spectral positions pertinent for the sonorant toward targets for the ensuing obstruent (e.g., a decrease in F_1 as the VT closes; F_2 falling for a labial obstruent or rising for an alveolar; F_3 rising for an alveolar and falling for a velar). In some cases, for a given formant, a smooth formant transition (obeying continuity) is clearly seen; when this happens, it is usually for F_3 or F_4, as these tend to be in the frequency range of overlap between sonorants and obstruents. Thus continuity constraints should vary with context, and not be applied across all frequencies equally.

Tracking formants at stop transitions is particularly interesting and difficult. It is important because crucial phonetic information is present during these brief periods, which are major factors in informing listeners of the articulation point of the stop. Other major phonetic cues are much more prominent in the speech signal, e.g., over longer durations and with greater intensity. Stops, on the other hand, do not cue their place during most of their duration, as the VT closure at that time means the only audible energy is the voice bar (if voiced).

The release of the VT occlusion at the end of a stop creates a brief (few ms) explosion of noise, which tends to excite all frequencies. Then, turbulent noise (frication) continues as the constriction opens for 10–40 ms, exciting the front cavities (usually F_2–F_4), as the VT moves toward the position for an ensuing sonorant. A velar constriction provides a long front cavity, with a low resonance near 2 kHz (F_2 or F_3). Velar resonances are higher due to a shorter front cavity. The spectrum of a labial burst is relatively flat and weak since there is essentially no front cavity to excite.

Most formant estimators either do poorly during obstruents, or simply claim that such regions do not need formant estimation. A statement such as the latter is relatively true, as many speech applications have a primary need for spectral estimation during the strong sections of speech; weaker sounds may often be modeled more simply, and their perception by listeners may be less critical to the communication task at hand. Nonetheless, knowing some details about what happens during the weaker sounds is of interest. Identifying the articulation point is mostly done via the formant transitions in adjacent sonorants, and the formant behavior at the stop release (normally considered to be part of the obstruent, and not part of the ensuing sonorant) is relevant. Furthermore, weak fricatives such as /v/ require attention.

11.6 Formant Estimation

Typical methods to estimate formants involve searching for peaks in spectral representations, usually from an STFT or LPC analysis [11.12, 13]. As LPC imposes an assumed simplified structure on the speech spectrum, it appears to have been employed most often in recent methods. In most LPC applications, one uses two poles per kilohertz of bandwidth (plus 2–4 additional poles to model other factors, such as the spectral tilt, which is due to glottal effects), on the assumption that speech contains one such formant in that range. A spectrum derived from an LPC model of N=10–16 poles is much more limited in variation (across frequency) than an STFT, which thus simplifies peak-picking. A disadvantage of using LPC is that its all-pole modeling is not perfect [11.14]; it chooses its pole positions to minimize mean-square error (MSE) for a fit of the speech to an N-pole spectral envelope. If the number of poles is not well chosen (e.g., to match the number of resonances clearly present in the speech), then the model spectrum is not as accurate as desired. For example, if there are too few poles in the model, the poles (selected via automatic analysis) have to place themselves in compromise locations between actual formants, and significant errors in formant tracking will result. Thus, it is rare that too few poles are used in LP analysis. Use of too many poles is a more likely risk; it is standard to follow the rule of thumb given above (2 poles/kHz), yet a given speech spectrum will often display fewer resonances than other phonemes (e.g., /u/ vowels and nasals often only show 2–3 formants, even in wideband applications, as higher formants tend to be very weak and thus poorly modeled via LPC, which focuses on peak energy). In this case, the extra poles (i. e., those not needed to directly model the actual resonances) locate themselves at non-formant frequencies to reduce the MSE further. Sometimes, if there are only, say, two extra poles, the additional poles will be real or have very wide bandwidth (to model the speech better via some broad spectral tilt effect). As LPC formant trackers usually examine the bandwidths of the LPC poles and reject (as potential formants) poles with wide bandwidth, such an effect is not a major problem. More-serious errors can arise when the additional poles model individual harmonics, e.g., in the strong F_1 region, as may happen in cases where individual formants have few harmonics; if F_1 has two dominant harmonics (e.g., the F_1 center frequency is located between two harmonics and the F_1 bandwidth is similar to F_0), then LPC may well assign four poles to model the two harmonics of F_1; this would lead to two formant candidates in the F_1 region, and require postprocessing to decide whether these candidates need to be merged into a single formant value.

Use of an STFT instead of an LPC spectrum avoids this latter problem, as it does not impose a specific (e.g., all-pole) model on the speech spectrum. Nonetheless, the issue of data reduction remains when using the STFT, as a typical STFT has 256–512 samples, corresponding to common FFT duration choices, for windows approximately 20–25 ms in length. One normally needs to smooth the STFT, and then do peak-picking. One can *pad with zeros*, i. e., select a much shorter range of speech samples than the FFT length (i. e., fewer samples than an estimated pitch period), to eliminate the harmonics from the spectral display, which effectively smooths the spectrum. In such a case, one normally needs to do pitch-synchronous analysis to choose at least the initial strong samples of each pitch period. Alternatively, one can smooth the FT of a full window with a low-pass filter operating in the frequency domain.

11.6.1 Continuity Constraints

It has generally been found that peak-picking methods need to be subject to continuity constraints so as to select from among multiple formant candidates, as there are often more candidates (i. e., spectral peaks) than formants [11.15]. One normally prunes away any candidate whose bandwidth is beyond the range of formants, i.e., a candidate whose bandwidth is less than 50 Hz, which is likely to be an interfering tone or an individual harmonic rather than a formant, or more than, say, 300 Hz, as formant bandwidths increase with center frequency, e.g., a roughly constant Q of around 5–6, so this upper threshold should increase with frequency. In sections of speech that appear to be sonorants, i. e., strong, voiced speech, a tracker aims to assign one formant to each possible range, i. e., roughly one formant per 1000 Hz. We need to allow for many individual cases where there are two formants below 1000 Hz (e.g., /o/ and /u/) and F_2 above 2 kHz (e.g., /i/), but within, say, the typical range of 0–4 kHz, there should be four formants (assuming a man's voice). If the signal has passed along a telephone line, then we should not expect to see more than three formants, as F_4 is often lost to the upper frequency cutoff of the phone lines (which preserve only the range of approximately 300–3200 Hz). Similarly, if the dynamic range of the spectral ampli-

tude obtained is limited, then F_4 may in general be too weak to be clearly observed, especially if the background noise level is high enough to obscure the weaker formants, which often include F_4.

The most difficult area for designing good continuity constraints is in the temporal dynamics. Automatic tracking of formants is difficult mostly owing to rapid changes in the formant patterns when a VT closure occurs or when the VT excitation changes state (between voiced and unvoiced); such changes often occur several times per second in speech. From frame to frame, in most speech, the formants generally change slowly. Other than the abrupt formant changes (which are due to major VT changes), the most rapid formant changes are normally seen in F_2 when the tongue moves quickly in lateral motion (e.g., /ai/ and /oi/) or when the lips round/unround. The maximum rate of change for a formant is approximately 20 Hz/ms (e.g., a change of 1200 Hz over 60 ms); so any proposed formant changes exceeding this threshold should be limited to major phoneme boundaries, where all the formants change and the overall intensity level also changes abruptly.

The apparent merging of formants has been noted as a major problem for formant trackers, i. e., when two or more formants are sufficiently close to each other to present an almost solid band of energy in a given spectral display. Many sounds have two formants close enough that they may potentially appear as one spectral peak (e.g., F_1–F_2 in /a/, /o/, and /u/, and F_2–F_3 in /i/ and /r/). One should not normally rely on small rises and falls (across frequency) in a spectrum to delineate individual formants, as such small changes can easily be due to window or harmonic effects. (An LPC spectrum, of course, normally does not have such rapid, small changes, but close formants also cause difficulties with LPC as well [11.14].) Thus, a spectral peak normally requires a significant rise in spectral amplitude over a frequency range on the order of a typical formant bandwidth, in order to be declared a good formant candidate.

It is generally by applying continuity constraints that one can resolve formant merges. While formants may be quite close during the steady state of a vowel, coarticulation with adjacent phonemes generally causes sufficient formant motion of all formants that nearby frames of speech clearly show separating tracks of spectral peaks. When a formant tracker is in doubt about a wide band of speech energy in a given section of speech, it can scan left and right (i. e., before and after) to look for a possible peak that may be moving away from the main band of energy. Often, this is a case of a weaker, rising F_3 or F_4 that was temporarily *merged* with F_2 or F_3, respectively; in recent papers on formant tracking, one often sees examples of the tracker incorrectly choosing a stronger and higher-frequency spectral peak as a formant, while ignoring a weaker track (a true formant) moving away from a wide band of speech energy (in which case all of these are formants).

A number of formant trackers focus on precise estimates of the center frequencies of resonances within frequency bands that are assumed to contain a single formant [11.16]. For each frame, they divide up the spectrum into such estimated bands in an initial analysis step, then use simpler estimation methods within each band. If indeed each band has a single resonance, the precise location of its center frequency and bandwidth is simpler through the use of adaptive bandpass filters that seek to isolate individual formants, thus allowing a more-precise focus on details of the (presumed single) resonance within each chosen band of frequency [11.16]. At first glance, this might seem to beg the question of formant tracking, as we have already identified the issue of separating formants that may closely approach each other as potentially difficult. Yet reasonable results appear possible with appropriate filtering, even in noisy conditions [11.17, 18]. One approach uses a parallel formant synthesizer, as was common in older TTS systems to generate a hypothesized synthetic speech spectrum (and thus with specific, known formants), and compares that spectrum to the actual speech spectrum, minimizing a spectral distance measure (often a quadratic cost function relating the two speech spectra, and also aiming for temporal continuity) via dynamic programming (DP, often via hidden Markov models) [11.19]. While DP is popular in formant trackers [11.11], some recent versions of such an approach seem not to need DP [11.20].

11.6.2 Use of Phase Shift

A common way to track formants is to estimate speech $S(z)$ in terms of a ratio of z-polynomials (e.g., an all-pole LPC spectrum), solve directly for the roots of the denominator, and identify each root as a formant if it has a narrow bandwidth at a reasonable frequency location [11.14]). Other approaches use related phase information [from $S(z)$] to decide whether a spectral peak is a formant. When one evaluates $S(z)$ along the unit circle $z = \exp(\mathrm{i}\omega)$, a large negative phase shift occurs when ω passes a pole close to the unit circle. As formants correspond to complex-conjugate pairs of poles with relatively narrow bandwidths (i. e., near the

unit circle), each spectral peak having such a phase shift is normally a formant. The phase shift approaches −180° for small formant bandwidths.

In cases where two formants appear as one broad spectral peak, a modified discrete Fourier transform DFT can resolve the ambiguity. The chirp z-transform (CZT) calculates the z-transform of the windowed speech on a contour inside the unit circle. Whereas the DFT samples $S(z)$ at uniform intervals on the unit circle, the CZT may take a spiral contour anywhere in the z-plane. It may be located near poles corresponding to a spectral peak of interest and thus need to be evaluated only for a small range of frequency samples in pertinent cases. As a contour can be much closer to the formant poles than for the DFT, the CZT can resolve two poles (for two closely spaced formants) into two spectral peaks. Since formant bandwidths tend to increase with frequency, the spiral contour often starts near $z = \alpha$, just inside the unit circle (e.g., $\alpha = 0.9$), and gradually spirals inward with increasing frequency $\omega_k = 2\pi k/N$ ($z_k = \alpha\beta^k \exp(\mathrm{i}\omega_k)$, with β just less than 1). This contour would thus follow the expected path of the formant poles and can eliminate problems of merged peaks in DFT displays.

Other recent formant tracking methods are also based on phase in related ways [11.21]. Formant trackers experience increased difficulty when F_0 exceeds formant bandwidths, e.g., $F_0 > 250\,\mathrm{Hz}$, as in children's voices. Harmonics in such speech are so widely separated that only one or two appear in each formant. As a result, spectral analyzers have a tendency to label the prominent harmonics as formants, which is generally wrong, as the center frequency of a formant is rarely an exact multiple of F_0.

11.6.3 Smoothing

Many pattern estimation algorithms, including formant estimators, need to do postprocessing on the raw data that comes out of the main estimation processing step. Usually, such schemes produce estimated values once per frame (e.g., every 5–10 ms in many speech applications); this applies to ASR and to speech coders, as well as to F_0 trackers. In ASR, estimated values for VT representations are modeled by probability distributions, and decisions are made on global probabilities combining hundreds (or more) of computations; a small deviation in any one parameter has little effect, so smoothing of individual parameters in a frame is rarely needed for ASR. Here, we assume that estimated formants may be used for a variety of applications, and thus we judge performance on how well each output value matches the actual speech, and thus are less tolerant of even small errors.

While formant decisions involve continuity constraints, these constraints normally do not greatly affect localized estimations; they are instead mostly used to avoid large errors (such as missing a formant entirely). The decision in each frame is usually based on the immediate speech window, whose placement and duration are rarely synchronized for optimal estimation results. As a result, estimated formant values are often noisy, in the sense that they err slightly above or below their actual values, owing to suboptimal calculations. The estimations are usually within perceivable ranges (i. e., if we were to synthesize speech with the estimated formant values, and compare this with synthesis based on the true values, listeners would hear no difference), but it is usually preferred to smooth such noisy formant signals to present a final formant contour that is both accurate and tidy.

One initial idea here would be simply to pass any noisy formant contour, as a time signal, through a low-pass filter, choosing the cutoff frequency of the filter so as to smooth the small random deviations and not the useful formant movements related to VT movements. However, as when smoothing the output of F_0 detectors, formant estimations sometimes change rapidly between individual frames, and applying a linear low-pass filter would produce a poor result when formants do actually change abruptly; instead of a true abrupt formant jump for a nasal, the smoothed pattern would show a gradual rise or fall.

Another difficulty with linear filtering is its behavior when mistakes occur in parameter extraction. Formant and F_0 estimators sometimes produce erroneous isolated estimates (i. e., for individual frame outputs) called *outliers*, which deviate greatly from the rest of the parameter contour. Such mistakes must be corrected in postprocessing. As linear filters give equal weight to all signal samples, they would propagate the effect of such a mistake into adjacent sections of the smoothed output formant contour.

Thus, a common alternative to linear filtering is median smoothing [11.1], which preserves sharp signal discontinuities while eliminating fine errors and gross outliers. Most smoothers (linear and nonlinear) operate on a finite time window of the input signal, but linear smoothers combine the windowed samples linearly to produce the smoothed output sample, whereas median smoothing chooses a single value from among

the window samples. In each data window, the samples are ordered in amplitude with no regard for timing within the window. The output sample is the median, i. e., the $[(N+1)/2]$th of N ordered samples (for odd N). Sudden discontinuities are preserved because no averaging occurs. Up to $(N-1)/2$ outlier samples, above or below the main contour, do not affect the output.

Median smoothers do well in eliminating outliers and in global smoothing, but do not provide very smooth outputs when dealing with noisy signals. Thus they are often combined with linear smoothers to yield a compromise smoothed output, whose sharp transitions are better preserved than with only linear filtering but with a smoother output signal than would be possible using only median smoothing.

11.7 Summary

This chapter has presented an introduction to formant tracking methods for speech. We have concentrated on the major approaches to formant tracking, generally using Fourier or LPC displays, and either peak-picking or solving for the roots in the LPC all-pole polynomial. There are also other proposed methods, such as a bank of inverse filters [11.22].

References

11.1 D. O'Shaughnessy: *Speech Communication: Human and Machine*, 2nd edn. (IEEE, Piscataway 2000)

11.2 J. Darch, B. Milner, S. Vaseghi: MAP prediction of formant frequencies and voicing class from MFCC vectors in noise, Speech Commun. **11**, 1556–1572 (2006)

11.3 R. Togneri, L. Deng: A state-space model with neural-network prediction for recovering vocal tract resonances in fluent speech from Mel-cepstral coefficients, Speech Commun. **48**(8), 971–988 (2006)

11.4 K. Weber, S. Ikbal, S. Bengio, H. Bourlard: Robust speech recognition and feature extraction using HMM2, Comput. Speech Lang. **17**(2–3), 195–211 (2003)

11.5 W. Ding, N. Campbell: Optimizing unit selection with voice source and formants in the CHATR speech synthesis system, Proc. Eurospeech (1997) pp. 537–540

11.6 J. Malkin, X. Li, J. Bilmes: A graphical model for formant tracking, Proc. IEEE ICASSP, Vol. 1 (2005) pp. 913–916

11.7 K. Sjlander, J. Beskow: WAVESURFER – an open source speech tool, Proc. ICSLP (2000)

11.8 L. Deng, L.J. Lee, H. Attias, A. Acero: A structured speech model with continuous hidden dynamics and prediction-residual training for tracking vocal tract resonances, Proc. IEEE ICASSP, Vol. 1 (2004) pp. 557–560

11.9 Y. Zheng, M. Hasegawa-Johnson: Formant tracking by mixture state particle filter, Proc. IEEE ICASSP, Vol. 1 (2004) pp. 565–568

11.10 D.T. Toledano, J.G. Villardebo, L.H. Gomez: Initialization, training, and context-cependency in HMM-based formant tracking, IEEE Trans. Audio Speech **14**(2), 511–523 (2006)

11.11 M. Lee, J. van Santen, B. Mobius, J. Olive: Formant tracking using context-dependent phonemic information, IEEE Trans. Speech Audio Process. **13**(5), 741–750 (2005), Part 2

11.12 S. McCandless: An algorithm for automatic formant extraction using linear prediction spectra, Proc. IEEE ICASSP **22**(2), 135–141 (1974)

11.13 G. Kopec: Formant tracking using hidden Markov models and vector quantization, Proc. IEEE ICASSP **34**(4), 709–729 (1986)

11.14 G.K. Vallabha, B. Tuller: Systematic errors in the formant analysis of steady-state vowels, Speech Commun. **38**(1–2), 141–160 (2002)

11.15 Y. Laprie, M.-O. Berger: Cooperation of regularization and speech heuristics to control automatic formant tracking, Speech Commun. **19**(4), 255–269 (1996)

11.16 K. Mustafa, I.C. Bruce: Robust formant tracking for continuous speech with speaker variability, IEEE Trans. Audio Speech **14**(2), 435–444 (2006)

11.17 A. Rao, R. Kumaresan: On decomposing speech into modulated components, IEEE Trans. Speech Audio Process. **8**(3), 240–254 (2000)

11.18 I.C. Bruce, N.V. Karkhanis, E.D. Young, M.B. Sachs: Robust formant tracking in noise, Proc. IEEE ICASSP, Vol. 1 (2002) pp. 281–284

11.19 L. Welling, H. Ney: Formant estimation for speech recognition, IEEE Trans. Speech Audio Process. **6**(1), 36–48 (1998)

11.20 B. Chen, P.C. Loizou: Formant frequency estimation in noise, Proc. IEEE ICASSP, Vol. 1 (2004) pp. 581–584

11.21 D.J. Nelson: Cross-spectral based formant estimation and alignment, Proc. IEEE ICASSP, Vol. 2 (2004) pp. 621–624

11.22 A. Watanabe: Formant estimation method using inverse-filter control, IEEE Trans. Speech Audio Process. **9**(4), 317–326 (2001)

9. Homomorphic Systems and Cepstrum Analysis of Speech

R. W. Schafer

In 1963, *Bogert*, *Healy*, and *Tukey* published a chapter with one of the most unusual titles to be found in the literature of science and engineering [9.1]. In this chapter, they observed that the logarithm of the power spectrum of a signal plus its echo (delayed and scaled replica) consists of the logarithm of the signal spectrum plus a periodic component due to the echo. They suggested that further spectrum analysis of the log spectrum could highlight the periodic component in the log spectrum and thus lead to a new indicator of the occurrence of an echo. Specifically they made the following observation:

> *In general, we find ourselves operating on the frequency side in ways customary on the time side and vice versa.*

As an aid in formalizing this new point of view, they introduced a number of paraphrased words. For example, they defined the *cepstrum* of a signal as the power spectrum of the logarithm of the power spectrum of a signal. (In fact, they used discrete-time spectrum estimates based on the discrete Fourier transform.) Similarly, the term *quefrency* was introduced for the independent variable of the cepstrum [9.1].

In this chapter we will explore why the cepstrum has emerged as a central concept in digital speech processing. We will start with definitions appropriate for discrete-time signal processing and develop some of the general properties and computational approaches for the cepstrum of speech. Using this basis, we will explore the many ways that the cepstrum has been used in speech processing applications.

9.1 Definitions

As stated above, *Bogert* et al. based their definition of the cepstrum on a rather loose interpretation of the power spectrum of an analog signal. Since effective application of the cepstrum concept required digital processing, it was necessary, early on, to develop a solid definition in terms of discrete-time signal theory [9.2, 3].

9.1.1 Definition of the Cepstrum

For discrete-time signals, a definition in the spirit of [9.1] is that the *cepstrum* of a signal is the inverse discrete-time Fourier transform (IDTFT) of the logarithm of the magnitude of the DTFT of the signal. That is, the cepstrum of a signal $x[n]$ is defined as

$$c[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} \log|X(\mathrm{e}^{\mathrm{i}\omega})|\,\mathrm{e}^{\mathrm{i}\omega n}\,\mathrm{d}\omega\,, \tag{9.1a}$$

where the DTFT of the signal is defined as [9.4],

$$X(\mathrm{e}^{\mathrm{i}\omega}) = \sum_{n=-\infty}^{\infty} x[n]\,\mathrm{e}^{-\mathrm{i}\omega n}\,. \tag{9.1b}$$

Equation (9.1a) is the definition of the cepstrum that we shall use throughout this chapter. Note that $c[n]$, being an IDTFT, is nominally a function of a discrete index n. If the input sequence is obtained by sampling an analog signal, i. e., $x[n] = x_\mathrm{a}(n/f_\mathrm{s})$, then it would be natural to associate *time* with the index n in the cepstrum. However, elaborating on the cepstrum conceit, *Bogert* et al. introduced the term *quefrency* for the name of the independent variable of the cepstrum [9.1]. This new term is often useful in describing the fundamental properties of the cepstrum. For example, low quefrencies correspond to slowly varying components in the log magnitude spectrum, while high quefrencies correspond to rapidly vary components of the log magnitude. Isolated peaks at multiples of a quefrency P_0 in the cepstrum correspond to a periodic component in the log magnitude with period $2\pi/P_0$ in normalized radian frequency ω or f_s/P_0 in cyclic analog frequency.

9.1.2 Homomorphic Systems

Contemporaneously with the introduction of the cepstrum concept, *Oppenheim* [9.5] developed a new theory of nonlinear systems that was based on the mathematical theory of linear vector spaces. The essence of this theory was that certain operations of signal combination (convolution and multiplication in particular) satisfy the same postulates as does vector addition in the theory of linear vector spaces. From this observation, Oppenheim showed that classes of nonlinear systems could be defined on the basis of a generalized principle of superposition. He termed such systems *homomorphic*. Of particular importance for our present discussion is the class of homomorphic systems for which the input and output are combined by convolution. Such systems are represented by the canonic form shown in Fig. 9.1, where the operator $D_*\{\ \}$ is called the *characteristic system for convolution* and $D_*^{-1}\{\ \}$ is its inverse. This characteristic system is defined by the property that when $x[n] = x_1[n] * x_2[n]$, the corresponding output is

$$\begin{aligned}\hat{x}[n] &= D_*\{x_1[n] * x_2[n]\}\\ &= D_*\{x_1[n]\} + D_*\{x_2[n]\}\\ &= \hat{x}_1[n] + \hat{x}_2[n]\,.\end{aligned} \tag{9.2}$$

That is, the characteristic system transforms a combination by convolution into a corresponding combination by addition.

The middle system in Fig. 9.1 is the system denoted $L\{\ \}$, which is an ordinary linear system satisfying the principle of superposition with addition as both the input and output operation for signal combination [9.4]. Note that Fig. 9.1 shows the input and output operations at the top and just outside of each block.

Finally, the inverse characteristic system must transform a sum into a convolution so that the overall system transformation satisfies $H\{x_1[n] * x_2[n]\} = H\{x_1[n]\} * H\{x_2[n]\} = y_1[n] * y_2[n]$.

From Fig. 9.1, all homomorphic systems for convolution differ only in the (ordinary) linear part $L\{\ \}$. The key to the definition is the characteristic system $D_*\{\ \}$, which turns convolution into addition, and as we will see, it is $D_*\{\ \}$ that makes the connection between the cepstrum and the theory of homomorphic systems for convolution. Indeed, Fig. 9.2 shows a sequence of mathematical operators that has the desired property in (9.2). Specifically, the output $\hat{x}[n]$ can be represented by the discrete-time Fourier transform equations

$$X(\mathrm{e}^{\mathrm{i}\omega}) = \sum_{n=-\infty}^{\infty} x[n]\,\mathrm{e}^{-\mathrm{i}\omega n} \tag{9.3a}$$

$$\hat{X}(\mathrm{e}^{\mathrm{i}\omega}) = \log[X(\mathrm{e}^{\mathrm{i}\omega})] \tag{9.3b}$$

$$\hat{x}[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} \hat{X}(\mathrm{e}^{\mathrm{i}\omega})\,\mathrm{e}^{\mathrm{i}\omega n}\,\mathrm{d}\omega\,. \tag{9.3c}$$

Equation (9.3a) and (9.3c) are the DTFT and IDTFT, respectively, while (9.3b) is the *complex* logarithm of the

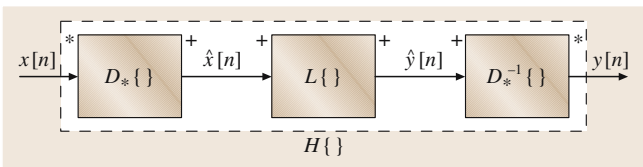


Fig. 9.1 Canonic form for homomorphic systems for convolution

complex function $X(e^{i\omega})$. It is easily seen that the cascade of these three operations has the desired property of turning convolution into addition. First, the DTFT of a convolution of two sequences is the product of their DTFTs [9.4]. Appropriately defined, the complex logarithm of a product of two functions is the sum of the complex logarithms of the individual functions. Finally, the IDTFT is a linear operator in the conventional sense, so the IDTFT of a sum is the sum of the corresponding inverse transforms [9.4]. (Interestingly, the DTFT is also linear in the conventional sense, but it is also a homomorphic operator with input operation convolution and output operation multiplication. Likewise, the IDTFT operator is both conventionally linear and homomorphic with input operation multiplication and output operation convolution.) Hence, if the complex logarithm in Fig. 9.2 is appropriately defined and computed, then if the input is $x[n] = x_1[n] * x_2[n]$, it follows that the output in in Fig. 9.2 is $\hat{x}[n] = \hat{x}_1[n] + \hat{x}_2[n]$.

The inverse of the characteristic system for convolution is depicted in Fig. 9.3. It is obtained by simply using the complex exponential to invert the effect of the complex logarithm.

The representation of Figs. 9.2 and 9.3 was developed by *Oppenheim*, *Schafer*, and *Stockham*, and first published in [9.2, 3]. Because of the obvious close relationship of the characteristic system for convolution as defined by (9.3a), (9.3b), and (9.3c) to the definition of the cepstrum in (9.1a) and (9.1b), they called the output of the characteristic system for convolution, the *complex cepstrum*. This is not because the complex cepstrum is complex; indeed, if $x[n]$ is real, then $\hat{x}[n]$ will also be real. Rather, the modifier *complex* is used to imply that the *complex logarithm* is used in the computation of the complex cepstrum. The complex logarithm in (9.3b) is defined as

$$\begin{aligned}\hat{X}(e^{i\omega}) &= \log\{X(e^{i\omega})\} \\ &= \log|X(e^{i\omega})| + i\arg\{X(e^{i\omega})\}\,, \end{aligned} \tag{9.4}$$

where if $X(e^{i\omega}) = X_1(e^{i\omega}) \cdot X_2(e^{i\omega})$, as for the convolution $x[n] = x_1[n] * x_2[n]$, then the following must hold:

$$\log|X(e^{i\omega})| = \log|X_1(e^{i\omega})| + \log|X_2(e^{i\omega})| \tag{9.5a}$$

$$\arg\left\{X(e^{i\omega})\right\} = \arg\{X_1(e^{i\omega})\} + \arg\left\{X_2(e^{i\omega})\right\}\,. \tag{9.5b}$$

Satisfying (9.5a) presents no serious difficulty, but (9.5b) is more problematic as we discuss below.

The relationship between the cepstrum and complex cepstrum can be obtained by noting that, if $x[n]$ is real,

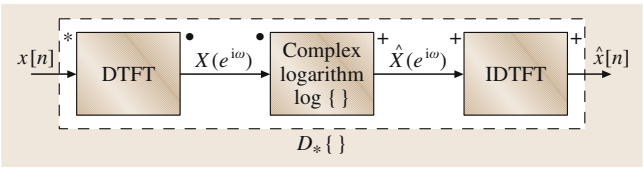


Fig. 9.2 DTFT representation of the characteristic system for convolution

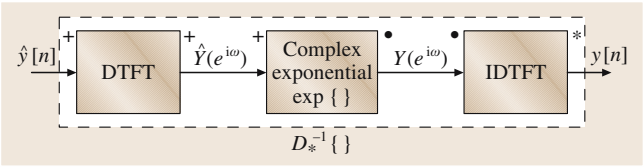


Fig. 9.3 DTFT representation of the inverse characteristic system for convolution

then $\log|X(e^{i\omega})|$ is a real and even function of ω, while $\arg\{X(e^{i\omega})$, the imaginary part of $\hat{X}(e^{i\omega})$, is a real and odd function of ω. This implies that [9.4],

$$c[n] = \mathrm{Ev}\{\hat{x}[n]\} = \frac{\hat{x}[n] + \hat{x}[-n]}{2}\,. \tag{9.6}$$

That is, as we have defined them, the cepstrum is the *even part* of the complex cepstrum.

9.1.3 Numerical Computation of Cepstra

So far, we have merely defined the cepstrum and complex cepstrum in terms of mathematical operators. To be useful for speech processing, we must replace these operators by computable operations. This can be done by noting that the discrete Fourier transform (DFT) [computed with a fast Fourier transform (FFT) algorithm] is a sampled (in frequency) version of the DTFT of a finite-length sequence, i. e., $X[k] = X(e^{i2\pi k/N})$ [9.4]. Figure 9.4 depicts the operations

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-i(2\pi k/N)n} \tag{9.7a}$$

$$\hat{X}[k] = \log|X[k]| + i\arg\{X[k]\} \tag{9.7b}$$

$$\tilde{\hat{x}}[n] = \frac{1}{N}\sum_{n=0}^{N-1} \hat{X}[k]\, e^{i(2\pi k/N)n}\,, \tag{9.7c}$$

where the DTFT has been replaced by the finite DFT computation. Note the *tilde* ($\tilde{}$) symbol above $c[n]$ and $\hat{x}[n]$ in Fig. 9.4 and in (9.7c), which is used to emphasize that the cepstra computed using the DFT suffer from time-domain aliasing due to the sampling of the log

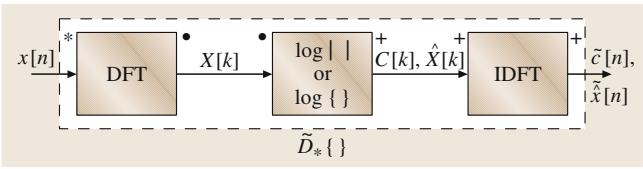


Fig. 9.4 Computing the cepstrum or complex cepstrum using the DFT

of the DTFT [9.4]. Specifically, the complex cepstrum computed by the DFT is related to the complex cepstrum defined by the DTFT by the equation

$$\tilde{\hat{x}}[n] = \sum_{r=-\infty}^{\infty} \hat{x}[n+rN] \,. \tag{9.8}$$

An identical equation holds for the time-aliased cepstrum $\tilde{c}[n]$.

The effect of time-domain aliasing can be made negligible by using a large value for N. A more-serious problem in computation of the complex cepstrum is the computation of the complex logarithm. As discussed above, we must compute samples of $\arg\{X(e^{i\omega})\} = \arg\{X_1(e^{i\omega}) \cdot X_2(e^{i\omega})\}$ such that

$$\arg\{X[k]\} = \arg\{X_1[k]\} + \arg\{X_2[k]\} \,, \tag{9.9}$$

and this requires special care. Standard arctan functions return the principal value of the phase angle, which we denote, $-\pi < \mathrm{ARG}\{X[k]\} \leq \pi$, and in general it must be assumed that

$$\mathrm{ARG}\{X[k]\} \neq \mathrm{ARG}\{X_1[k]\} + \mathrm{ARG}\{X_2[k]\} \,, \tag{9.10}$$

when $X[k] = X_1[k]X_2[k]$. It is not that the principal value gives the wrong phase for the product of two DFTs, but that it is not equal to the sum of the two individual principal value phases. The problem is the discontinuities that result from the modulo 2π computation. If the principal value phase can be *unwrapped* by finding a sequence $R[k]$ such that

$$\arg\{X[k]\} = \mathrm{ARG}\{X[k]\} + 2\pi R[k] \,, \tag{9.11}$$

then the resulting phase would have the desired property of (9.9). The sequence $R[k]$ will have integer values that are constant over subintervals of $0 \leq k < N/2$. Abrupt changes will occur at the *discontinuities* of $\mathrm{ARG}\{X[k]\}$. The earliest phase unwrapping algorithm for cepstrum computation took the straightforward approach of searching for discontinuities of size 2π in $\mathrm{ARG}\{X[k]\}$ [9.2,3]. This approach requires a very small frequency sample spacing (N large) so as not to miss discontinuities that occur when the phase has a steep slope. *Tribolet* gave an improved algorithm that uses both the principal value phase and the phase derivative (which can be computed directly from $X[k]$) [9.6].

While phase unwrapping can be problematic in some situations involving the complex cepstrum, it is not a problem in computing the cepstrum, where it is not used, and as we will see in the next section, phase unwrapping can be avoided by using the z-transform.

9.2 Z-Transform Analysis

The characteristic system for convolution can also be represented by the two-sided z-transform, as depicted in Fig. 9.5. This representation is very useful for theoretical investigations, and recent developments in polynomial root finding have made the z-transform representation a viable computational basis as well. For this purpose, we assume that the input signal $x[n]$ has a rational z-transform of the form

$$X(z) = X_{\max}(z) \cdot X_{\mathrm{uc}}(z) \cdot X_{\min}(z) \,, \tag{9.12}$$

where

$$\begin{aligned} X_{\max}(z) &= z^{M_o} \prod_{k=1}^{M_o} \left(1 - a_k z^{-1}\right) \\ &= \prod_{k=1}^{M_o} (-a_k) \prod_{k=1}^{M_o} \left(1 - a_k^{-1} z\right) , \end{aligned} \tag{9.13a}$$

$$X_{\mathrm{uc}}(z) = \prod_{k=1}^{M_{\mathrm{uc}}} \left(1 - e^{i\theta_k} z^{-1}\right) , \tag{9.13b}$$

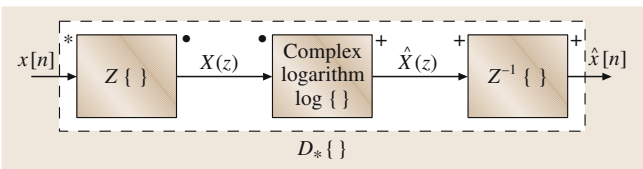


Fig. 9.5 z-transform representation of the characteristic system for convolution

$$X_{\min}(z) = A \frac{\prod_{k=1}^{M_i} \left(1 - b_k z^{-1}\right)}{\prod_{k=1}^{N_i} \left(1 - c_k z^{-1}\right)} \,. \tag{9.13c}$$

The zeros of $X_{\max}(z)$, i. e., $z_k = a_k$, are zeros of $X(z)$ outside of the unit circle ($|a_k| > 1$). The factor $X_{\max}(z)$ is thus the *maximum-phase* part of $X(z)$. The factor $X_{\text{uc}}(z)$ contains all the zeros (with angles θ_k) *on* the unit circle. The *minimum-phase* part is $X_{\min}(z)$, where b_k and c_k are zeros and poles, respectively, that are inside the unit circle ($|b_k| < 1$ and $|c_k| < 1$). The factor z^{M_o} implies a shift of M_o samples to the left. It is included to simplify the results in (9.16).

The complex cepstrum of $x[n]$ is determined by assuming that the complex logarithm $\log[X(z)]$ results in the sum of logarithms of each of the product terms, i. e.,

$$\begin{aligned}\hat{X}(z) = {} & \log\left|\prod_{k=1}^{M_o}(-a_k)\right| + \sum_{k=1}^{M_o} \log\left(1 - a_k^{-1} z\right) \\ & + \sum_{k=1}^{M_{\text{uc}}} \log(1 - \mathrm{e}^{\mathrm{i}\theta_k} z^{-1}) \\ & + \log|A| + \sum_{k=1}^{M_i} \log\left(1 - b_k z^{-1}\right) \\ & - \sum_{k=1}^{N_i} \log\left(1 - c_k z^{-1}\right) \,. \end{aligned} \tag{9.14}$$

Applying the power series expansion

$$\log(1-a) = -\sum_{n=1}^{\infty} \frac{a^n}{n} \,, \quad |a| < 1 \tag{9.15}$$

to each of the terms in (9.14) and collecting the coefficients of the positive and negative powers of z gives

$$\hat{x}[n] = \begin{cases} \displaystyle\sum_{k=1}^{M_o} \frac{a_k^n}{n} & n < 0 \\ \displaystyle \log|A| + \log\left|\prod_{k=1}^{m_o}(-a_k)\right| & n = 0 \\ \displaystyle \left(-\sum_{k=1}^{M_{\text{uc}}} \frac{\mathrm{e}^{\mathrm{i}\theta_k n}}{n} \right. & \\ \displaystyle \left. -\sum_{k=1}^{M_i} \frac{b_k^n}{n} + \sum_{k=1}^{N_i} \frac{c_k^n}{n}\right) & n > 0 \end{cases} \,. \tag{9.16}$$

Given all the poles and zeros of a z-transform $X(z)$, (9.16) allows us to compute a finite set of samples of the complex cepstrum with no approximation. This is clearly the case in theoretical analysis where the poles and zeros are specified, but (9.16) is also useful as the basis for computation. All that is needed is a process for obtaining the z-transform as a rational function and a process for finding the zeros of the numerator and denominator. This has become more feasible with increasing computational power and with new algorithmic advances in finding roots of large polynomials [9.7].

One method of obtaining a z-transform is simply to select a finite-length sequence of samples of a signal. The z-transform is then simply a polynomial with the samples $x[n]$ as coefficients, i. e.,

$$\begin{aligned} X(z) &= \sum_{n=0}^{M} x[n] z^{-n} \\ &= A \prod_{k=1}^{M_o} \left(1 - a_k z^{-1}\right) \prod_{k=1}^{M_i} \left(1 - b_k z^{-1}\right) \,. \end{aligned} \tag{9.17}$$

A second method that yields a z-transform is the method of linear predictive analysis, which we will discuss briefly in Sect. 9.5.

9.3 Discrete-Time Model for Speech Production

A short segment of a sampled speech waveform is shown in Fig. 9.6. Note that about the first 90 ms (720 samples at 8000 samples/s) look like random noise, while the remaining samples in view appear to be almost periodic. Such a speech signal is the output of a physical system that produces an acoustic wave whose properties vary in time to encode a message. To create a speech signal, humans perform a coordinated sequence of physical gestures involving the lungs, vocal cords, tongue, and lips. Assuming that we have a sampled speech signal, it is helpful to have a discrete-time system model upon which to base analysis [9.8, 9]. A simple model is depicted in Fig. 9.7. The block on the right labeled 'time-varying digital filter' represents a linear system with a slowly time-varying frequency response (or impulse response). Its purpose is to model the frequency

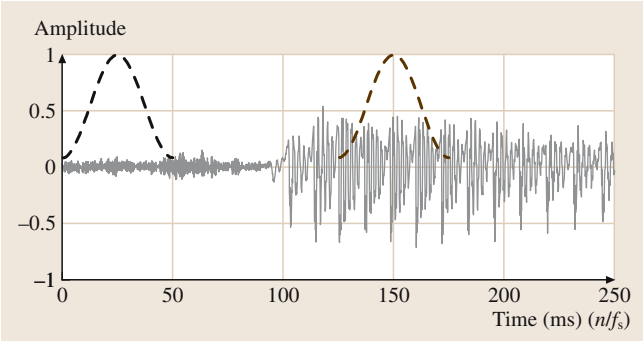


Fig. 9.6 Segment of sampled speech waveform (samples connected by straight lines for plotting)

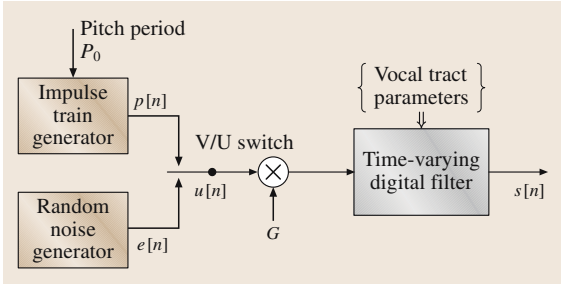


Fig. 9.7 Simplified discrete-time model for the speech signal

resonances (formants) of the human vocal tract. It is common to assume that, over short time intervals, the system is not time-varying, and that it is characterized by an all-pole rational system function of the form

$$H(z) = \frac{G}{1 - \sum_{k=1}^{p} \alpha_k z^{-k}} = \frac{G}{\prod_{k=1}^{p} \left(1 - c_k z^{-1}\right)} , \quad (9.18)$$

where $|c_k| < 1$ for stability. The blocks labeled 'impulse train generator' and 'random noise generator' represent the glottal pulse excitation and fricative (constriction) excitation of the vocal tract, respectively. The multiplicative factor G is lumped with $H(z)$ in (9.18), but it could be considered part of the excitation since it simply applies a gain to the excitation. By properly choosing the parameters of the excitations and the linear system, it is possible to create discrete-time signals that are perceived to be very close, if not identical, to a given sampled human-produced speech signal. Therefore, the model of Fig. 9.7 is the basis for most analysis/synthesis speech coders and automatic speech recognition systems, and much work has been done on estimating the time-varying parameters of this model from sampled speech signals. Homomorphic filtering and cepstrum analysis are important techniques for such analysis.

9.4 The Cepstrum of Speech

A fundamental tenet of digital speech processing is that the properties of the speech signal change slowly relative to the sampling rate of the signal, and furthermore, that these properties can be *sampled* by a process of *short-time analysis* in which speech properties such as pitch and vocal tract response are assumed to hold constant over a short time interval (called a window or frame) of duration on the order of 20–30 ms. This is illustrated in Fig. 9.6 by the 401-sample signal analysis windows placed at sample index 0 and 1000 (0 and 125 ms). Typically for the 8000 samples/s sampling rate of the speech in Fig. 9.6, the analysis window would be moved in steps of 80–160 samples (10–20 ms).

9.4.1 Short-Time Cepstrum of Speech

Following [9.9], we assume that the model of Fig. 9.7 represents the sampled speech signal $s[n]$, and that over the analysis interval the output of the model equals the speech signal. Furthermore, over the length of the window L, we assume that

$$s[n] = u[n] * h[n] , \quad 0 \le n \le L-1 , \quad (9.19)$$

where $h[n]$ is the impulse response corresponding to the system function in (9.18). The impulse response $h[n]$ models the combined effects of the gain G, the vocal-tract frequency response, the glottal wave shape, and radiation of sound at the lips, but we will refer to it simply as the *vocal-tract impulse response*. Also, we assume that the impulse response $h[n]$ is short compared to the window so that the windowed segment can be represented as

$$\begin{aligned} x[n] &= w[n](u[n] * h[n]) \\ &\approx u_w[n] * h[n] , \quad 0 \le n \le L-1 , \end{aligned} \quad (9.20)$$

where $u_w[n] = w[n]u[n]$, i. e., any tapering due to the analysis window is incorporated into the excitation. In the case of voiced speech, $u[n]$ is an impulse train of the form

$$u[n] = p[n] = \sum_{k=0}^{N_p-1} \delta[n - kP_0] \,, \tag{9.21}$$

where $\delta[n]$ represents the unit sample sequence and P_0 is the discrete-time *pitch period*. In the case of unvoiced speech, the excitation would be a white noise sequence, which is spectrally shaped by the linear filter.

For voiced speech, the windowed excitation is

$$u_w[n] = w[n]p[n] = \sum_{k=0}^{N_p-1} w_{P_0}[k]\delta[n - kP_0] \,, \tag{9.22}$$

where $w_{P_0}[k]$ is the *time-sampled* window sequence defined as

$$w_{P_0}[k] = \begin{cases} w[kP_0] & k = 0, 1, \ldots, N_p - 1 \\ 0 & \text{otherwise} \end{cases} . \tag{9.23}$$

From (9.22), the DTFT of $u_w[n]$ is

$$U_w(\mathrm{e}^{\mathrm{i}\omega}) = \sum_{k=0}^{N_p-1} w_{P_0}[k]\mathrm{e}^{-\mathrm{i}\omega k P_0} = W_{P_0}(\mathrm{e}^{\mathrm{i}\omega P_0}) \,, \tag{9.24}$$

and from (9.24) it follows that $U_w(\mathrm{e}^{\mathrm{i}\omega})$ is periodic in ω with period $2\pi/P_0$. Therefore,

$$\hat{X}(\mathrm{e}^{\mathrm{i}\omega}) = \log\{H(\mathrm{e}^{\mathrm{i}\omega})\} + \log\left\{U_w(\mathrm{e}^{\mathrm{i}\omega})\right\} \tag{9.25}$$

has two components: (i) $\log\{H(\mathrm{e}^{\mathrm{i}\omega})\}$, due to the vocal-tract frequency response and slowly varying in ω, and (ii) $\log\{W_{P_0}(\mathrm{e}^{\mathrm{i}\omega P_0})\}$, due to the excitation and periodic with period $2\pi/P_0$. (For signals sampled with sampling rate f_s, this period corresponds to f_s/P_0 Hz in cyclic analog frequency.) The complex cepstrum of the windowed speech segment $x[n]$ is therefore

$$\hat{x}[n] = \hat{h}[n] + \hat{u}_w[n] \,. \tag{9.26}$$

For voiced speech, the cepstral component due to the excitation has the form

$$\hat{u}_w[n] = \begin{cases} \hat{w}_{P_0}[n/P_0] & n = 0, \pm P_0, \pm 2P_0, \ldots \\ 0 & \text{otherwise} \end{cases} . \tag{9.27}$$

(Note that $\hat{w}_{P_0}[n]$ corresponds to $\log\{W_{P_0}(\mathrm{e}^{\mathrm{i}\omega})\}$, so through the upsampling theorem [9.4], $\log\{W_{P_0}(\mathrm{e}^{\mathrm{i}\omega P_0})\}$ corresponds to (9.27).) That is, in keeping with the $2\pi/P_0$ periodicity of $\log\{U_w(\mathrm{e}^{\mathrm{i}\omega})\} = \log\{W_{P_0}(\mathrm{e}^{\mathrm{i}\omega P_0})\}$, the corresponding complex cepstrum (or cepstrum) has impulses (isolated samples) at quefrencies that are multiples of P_0. For unvoiced speech, no such periodicity occurs in the logarithm, and therefore no cepstral peaks occur.

To illustrate the cepstrum analysis of speech, a 401-sample segment of voiced speech was selected by a Hamming window positioned at sample 1000 in Fig. 9.6. The resulting windowed $x[n]$ is shown in Fig. 9.8a. The log magnitude and unwrapped phase for $X(\mathrm{e}^{\mathrm{i}\omega})$ are shown as the rapidly oscillating curves in Figs. 9.9a and 9.9b, respectively. The resulting complex cepstrum is shown in Fig. 9.10a and the cepstrum is shown in Fig. 9.10b, i. e., Fig. 9.10a is the IDTFT of $\log\{X(\mathrm{e}^{\mathrm{i}\omega})\}$ with Fig. 9.9a as the real part and Fig. 9.9b as the imaginary part, while Fig. 9.10b is the IDTFT of $\log|X(\mathrm{e}^{\mathrm{i}\omega})|$ with zero imaginary part. [From (9.6), Figure 9.10b is also the even part of Fig. 9.10a.]

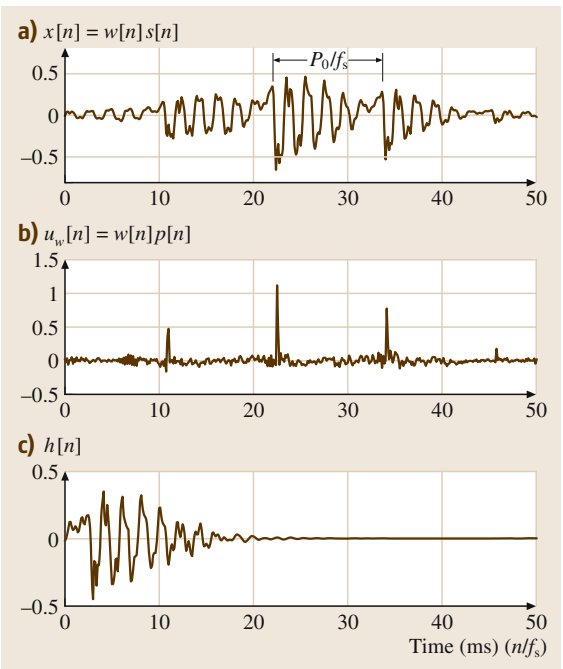


Fig. 9.8a–c Time waveforms: **(a)** windowed speech segment, **(b)** output corresponding to high-quefrency components of the complex cepstrum, and **(c)** output corresponding to low-quefrency components of the complex cepstrum

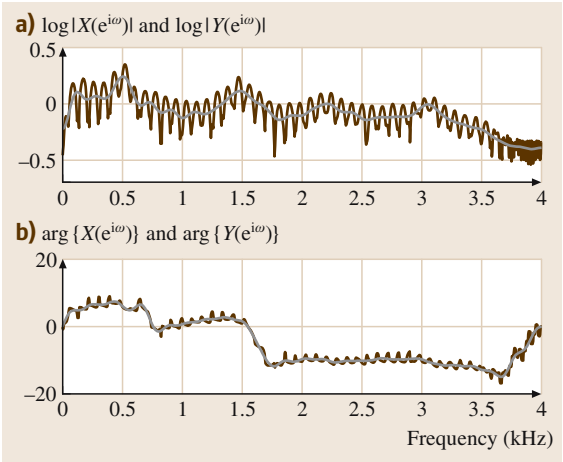


Fig. 9.9 **(a)** Log magnitude (the *light-gray curve* corresponds to low-quefrency components), and **(b)** unwrapped phase

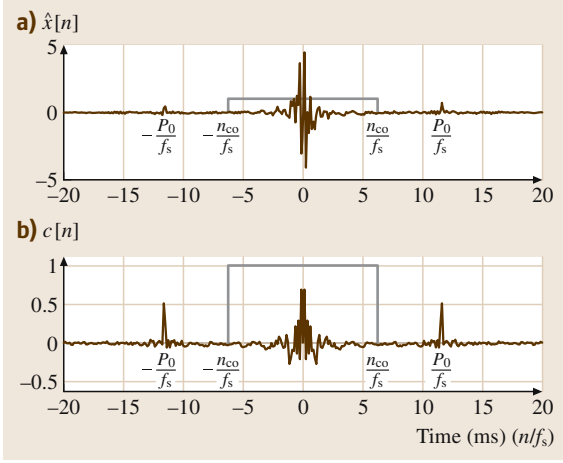


Fig. 9.10 **(a)** Complex cepstrum. **(b)** Cepstrum

Observe the rapidly varying periodic components in Fig. 9.9a,b, and the corresponding peaks in Fig. 9.10 located at about quefrency 11.5 ms ($P_0 = 92$ samples at $f_s = 8000$ samples/s rate). These peaks in the cepstrum and complex cepstrum are due to the voiced nature of the speech segment, and their quefrency locations correspond to the *pitch period* in Fig. 9.8a. Furthermore, note that when expressed in Hz, the period of the ripples in Fig. 9.9 is $8000/92 = 87$ Hz. Thus, we see from the cepstrum that the fundamental voice frequency for this segment of speech is about 87 Hz.

9.4.2 Homomorphic Filtering of Speech

Homomorphic filtering can be used to separate the components of the speech model. In the terminology of Bogert et al., we *lifter* the signal by multiplying its complex cepstrum (or cepstrum) by a sequence $\ell[n]$, i. e., we form

$$\hat{y}[n] = \ell[n]\hat{x}[n] \,. \tag{9.28}$$

This is one choice for the linear system $L\{\ \}$ shown in the canonic system of Fig. 9.1. The corresponding *liftered output* is obtained by implementing the inverse characteristic system of Fig. 9.3. Note that the corresponding operation on the complex logarithm is periodic convolution in the frequency domain, i. e.,

$$\hat{Y}(\mathrm{e}^{\mathrm{i}\omega}) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \hat{X}(\mathrm{e}^{\mathrm{i}\theta})\hat{L}(\mathrm{e}^{\mathrm{i}(\omega-\theta)})\,\mathrm{d}\theta \,. \tag{9.29}$$

In words, liftering is equivalent to *frequency-invariant* linear filtering of the complex logarithm. For example, multiplication by a function such as the light-gray curve in Fig. 9.10a would have the effect of removing the periodic component from the log magnitude and unwrapped phase in Fig. 9.9a or b or in the case of the cepstrum, the periodic component would be removed from only the log magnitude. Specifically, if we multiply all components of the complex cepstrum of Fig. 9.10a above quefrency 50 (6.25 ms) by zero, and compute the DTFT of the result, we obtain $\hat{Y}(\mathrm{e}^{\mathrm{i}\omega})$, as shown by the smooth curves in Fig. 9.9a,b. If we send the resulting *liftered* complex logarithm through the exponential and inverse DTFT of the inverse characteristic system of Fig. 9.3, the result is as shown in Fig. 9.8c, which we can interpret as an estimate of $h[n]$. Similarly, we can remove the low quefrencies by retaining cepstrum components above some cutoff quefrency, and we obtain an output that depends mainly on the excitation components of the complex cepstrum. Such a result is shown in Fig. 9.8b, which is an estimate of $u_w[n]$. Notice that the impulses at multiples of the pitch period retain the shape of the Hamming window as suggested by (9.22).

The previous example illustrates that homomorphic filtering can be used to separate the components of a convolution, and this technique can be used to *deconvolve* short segments of voiced speech. Specific applications of this approach will be considered in the following sections.

9.5 Relation to LPC

The technique of linear predictive analysis, usually referred to as linear predictive coding (LPC), is widely used in digital speech processing systems. It is therefore of interest to consider the relationship between cepstrum analysis and linear predictive analysis. In LPC, an all-pole model such as (9.18) is obtained by first assuming that the model output $s[n]$ is equal to the given speech signal $x[n]$, and then computing a set of predictor coefficients α_k that minimize the mean-squared prediction error

$$E = \left\langle e^2[n] \right\rangle = \left\langle \left(x[n] - \sum_{k=1}^{p} \alpha_k x[n-k] \right)^2 \right\rangle . \quad (9.30)$$

The optimum coefficients satisfy the set of p linear equations in the p unknown coefficients α_k

$$\sum_{k=1}^{p} \alpha_k r[|k-i|] = r[i] \; i = 1, 2, \ldots , p , \quad (9.31a)$$

and the gain constant G is given by

$$G = r[0] - \sum_{k=1}^{p} \alpha_k r[k] , \quad (9.31b)$$

where $r[k] =< x[n]x[n+k] >$ with $< >$ denoting averaging over a finite time window. Thus, $p+1$ values of the autocorrelation function $r[k]$ are sufficient to determine all the $p+1$ parameters of the all-pole model of (9.18) [9.8].

9.5.1 LPC Versus Cepstrum Smoothing

Figure 9.11 shows the autocorrelation function and the cepstrum of the segment of speech in Fig. 9.8a. (As is common in LPC analysis [9.8], the high frequencies were pre-emphasized by a first difference operator prior to LPC and cepstral analysis.) Note that the autocorrelation function $r[k]$ is the IDTFT of $|X(e^{i\omega})|^2$, while the cepstrum $c[n]$ is the IDTFT of $\log|X(e^{i\omega})|$, so it is not surprising that they should be related and have similar properties. Like the cepstrum, the autocorrelation shows a peak at the time corresponding to the pitch period of the speech signal; however, this peak is not nearly as distinct as the peak in the cepstrum. In LPC analysis, the periodic variations of the short-time Fourier transform are removed by taking p to be much less than the pitch period P_0. The faint gray line in Fig. 9.11a shows the autocorrelation values selected to determine a system model with $p = 12$, which is much less than the value $P_0 = 92$ determined by cepstrum analysis to be the period of the segment of speech under analysis. Such a value for p is sufficient to represent the combined effects of the vocal tract, glottal spectrum, and radiation at the lips for a sampling rate of 8 kHz, but too small to represent the fine detail in the short-time spectrum. If $H(z)$ in (9.18) is evaluated on the unit circle of the z-plane, we get an estimate of the frequency response of the linear system of the speech model, i. e., the function $\log|H(e^{i\omega/f_s})|$, which is plotted in Fig. 9.12 as the solid curve. It is superimposed with the original $\log|X(e^{i\omega/f_s})|$, shown as the rapidly varying light-gray plot.

For comparison, the dashed line is the result of liftering the short-time spectrum with a cepstrum cutoff quefrency of $n_{co} = 50$ samples and the dash-dot line shows liftering with a lower cutoff quefrency of $n_{co} = 13$. Thus, both the LPC model and cepstrum liftering can smooth out the periodic component of the short-time spectrum of a segment of voiced speech. Notice that the peaks of the LPC spectrum tend be biased toward a nearby harmonic peak of the short-time spectrum. This can cause undesirable variability in LPC representations. Generally, the LPC order p is chosen to provide one resonance (two complex-conjugate poles) per kilohertz of bandwidth with 2–4 extra poles to represent overall spectral tilt, etc. The liftered spectra, on the other hand, are essentially local averages of the log of the short-time Fourier transform magnitude. Thus, they follow the slow variations of the log of the short-time spectrum, and the lower the cutoff quefrency, the smoother the resulting spectrum estimate.

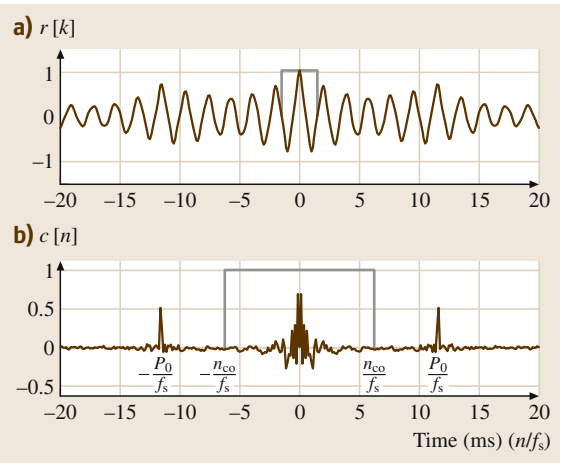


Fig. 9.11 **(a)** Autocorrelation function showing points used in LPC analysis. **(b)** Cepstrum showing points used in homomorphic smoothing of the short-time spectrum

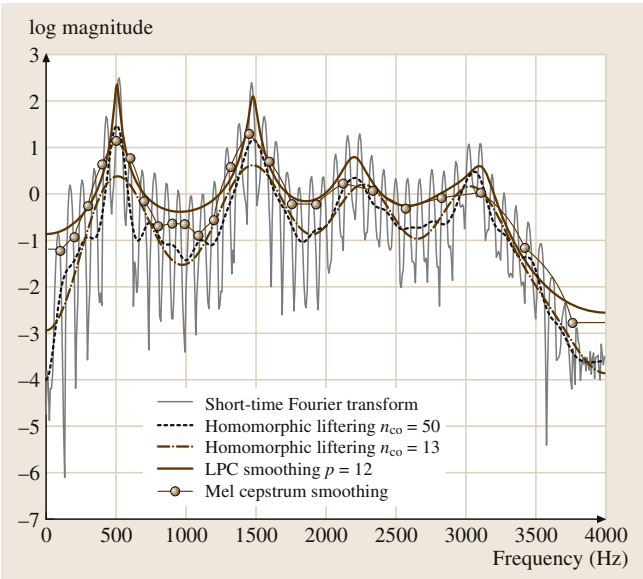


Fig. 9.12 Comparison of spectral smoothing methods

9.5.2 Cepstrum from LPC Model

In the LPC method, a short segment of windowed speech is represented first by $p+1$ autocorrelation values $r[k]$, $k = 0, 1, \ldots, p$, then by G and the predictor coefficients α_k, $k = 1, 2, \ldots, p$, and finally by the z-transform $H(z)$ given by (9.18). The truncation of the autocorrelation function to $p+1$ values causes the smoothing that we seek in estimating the vocal tract system. If we realize that, corresponding to $H(z)$, there is an impulse response $h[n]$, then it follows that the LPC analysis has computed the parameters of the z-transform of $h[n]$, and thus, we can think of LPC analysis as an implementation of the Z{ } operator in Fig. 9.5 when the input is $h[n]$. In Sect. 9.2, we showed that, given the rational z-transform of a sequence, we can compute the complex cepstrum directly from the zeros of the numerator and denominator polynomials. From (9.18) and (9.16) and the fact that the LPC model estimate is minimum phase (all poles inside the unit circle) [9.8], it follows that the complex cepstrum of the impulse response of the LPC model is

$$\hat{h}[n] = \begin{cases} 0 & n < 0 \\ \log G & n = 0 \\ \sum_{k=1}^{M_i} \frac{c_k^n}{n} & n > 0 \end{cases} \tag{9.32}$$

and the cepstrum would be

$$c[n] = \mathrm{Ev}\{\hat{h}[n]\} = \begin{cases} \log G & n = 0 \\ \frac{1}{2} \sum_{k=1}^{M_i} \frac{c_k^{|n|}}{|n|} & n \neq 0 \end{cases} . \tag{9.33}$$

9.5.3 Minimum Phase and Recursive Computation

Minimum-phase sequences like the LPC model impulse response have a number of important properties. Some of these properties are themselves sufficient to define the minimum-phase condition. For example, all the poles and zeros of the z-transform of a minimum-phase sequence are inside the unit circle. An equivalent statement is that the complex cepstrum is causal, i. e., $\hat{h}[n] = 0$ for $n < 0$ as in (9.32).

The special properties of a minimum-phase sequence can be used to derive additional computational algorithms that can be useful in speech processing. For example, using the derivative theorem for z-transforms [9.4] and the causality of minimum-phase sequences and their complex cepstra, it can be shown [9.3] that the impulse response of a minimum-phase system and its corresponding complex cepstrum are related by the recursion formula

$$\hat{h}[n] = \begin{cases} 0 & n < 0 \\ \log G & n = 0 \\ \frac{h[n]}{h[0]} - \sum_{k=1}^{n-1} \left(\frac{k}{n}\right) \frac{\hat{h}[k]h[n-k]}{h[0]} & n \geq 1 \end{cases} \tag{9.34}$$

If we write the system function of the LPC model of (9.18) as

$$H(z) = \frac{G}{1 - \sum_{k=1}^{p} \alpha_k z^{-k}} = \frac{G}{A(z)} , \tag{9.35}$$

we can define

$$a[n] = \begin{cases} 1 & n = 0 \\ -\alpha_n & 1 \leq n \leq p \\ 0 & \text{otherwise} \end{cases} \tag{9.36}$$

as the sequence whose z-transform is the denominator polynomial $A(z)$. Since the complex cepstrum is the

inverse z-transform of $\hat{H}(z) = \log G - \log\{A(z)\}$, it follows that $\hat{h}[n] = (\log G)\delta[n] - \hat{a}[n]$, and using (9.34) and the fact that $\hat{h}[n] = -\hat{a}[n]$ for $n \geq 1$, we can express the complex cepstrum $\hat{h}[n]$ directly in terms of the predictor coefficients as

$$\hat{h}[n] = \begin{cases} 0 & n < 0 \\ \log G & n = 0 \\ \alpha_n + \sum_{k=1}^{n-1} \left(\frac{k}{n}\right) \hat{h}[k]\alpha_{n-k} & n > 0 \end{cases} , \quad (9.37)$$

From (9.37) we see that the predictor coefficients can be obtained from the complex cepstrum through

$$\alpha_n = \hat{h}[n] - \sum_{k=1}^{n-1} \left(\frac{k}{n}\right) \hat{h}[k]\alpha_{n-k} \quad 1 \leq n \leq p \, . \quad (9.38)$$

From (9.38), it follows that $p+1$ values of the complex cepstrum are sufficient to fully determine the LPC model since all the predictor coefficients and G can be computed from $\hat{h}[n]$ for $n = 0, 1, \ldots, p$.

Kopec et al. [9.10] showed several other ways that linear prediction and cepstrum analysis can be combined to separate the components of more-complicated models than the all-pole model usually assumed in LPC analysis.

9.6 Application to Pitch Detection

The cepstrum was first applied in the speech processing field to determining the excitation parameters for the discrete-time speech model of Fig. 9.7. According to *Noll* [9.11], his colleague M. R. Schroeder suggested to him that the periodic structure of voiced speech was very similar to the echo structure that motivated their Bell Telephone Laboratories colleagues to define the cepstrum. He applied the cepstrum using short-time Fourier analysis as we have described in Sect. 9.4.1, and like *Bogert* et al. [9.1] he was only interested in *detecting* periodicity, not in extracting the details of the speech model. Noll proposed a pitch-detection algorithm that was based on cepstrum analysis implemented as in Fig. 9.4 and applied to successive short-time windowed segments of a speech signal. This is illustrated in Fig. 9.6 for two window positions, one clearly unvoiced speech and the later one voiced speech.

Figure 9.13 shows a plot that is very similar to the plot first published by *Noll* [9.11]. On the left is a sequence of log short-time spectra (rapidly varying curves) and on the right is the corresponding sequence of cepstra computed from the log spectra on the left. Spectrum and cepstrum number 1 are from the segment of speech selected by the leftmost window in Fig. 9.6. The successive spectra and cepstra are for 50 ms segments obtained by moving the window in steps of 12.5 ms (100 samples at 8000 samples/s). Thus, the second window position in Fig. 9.6 corresponds to spectrum and cepstrum number 11 in Fig. 9.13. From the given window length and time increment, we can see from Fig. 9.6 that for positions 1–5, the window will only include unvoiced speech, while for positions 6 and 7 the signal will be partly voiced and partly unvoiced. For positions 8–15 the window only includes voiced speech. Note that the rapid variations of the unvoiced spectra appear random with no periodic structure. This is typical of Fourier transforms (periodograms) of short segments of random signals. On the other hand, the spectra for voiced segments have a structure of periodic ripples due to the harmonic structure of the quasiperiodic segment of voiced speech. As can be seen from the plots on the right, the cepstrum peak at about quefrency 11–12 ms strongly signals voiced speech. As we have shown, the quefrency of the peak is an accurate estimate of the pitch period during the corresponding speech interval. As shown in Fig. 9.11, the autocorrelation function also displays an indication of periodicity, but not not nearly as clearly as does the cepstrum.

The essence of the pitch-detection algorithm proposed by Noll is to compute a sequence of short-time cepstra and search each successive cepstrum for a peak in the quefrency region of the expected pitch period. The presence of a strong peak implies voiced speech, and the quefrency location of the peak gives an estimate of the pitch period. As in most signal processing applications such as this, the algorithm includes many features designed to handle cases that do not fit the underlying model very well. For example, for frames 6 and 7 the cepstrum peak is weak, corresponding to the transition from unvoiced to voiced speech. In other problematic cases, the peak at twice the pitch period may be stronger than the peak at the quefrency of the pitch period. Noll applied temporal continuity constraints to prevent such errors. The reader should consult *Noll*'s paper [9.11] for the full details of his cepstrum pitch-detection algorithm.

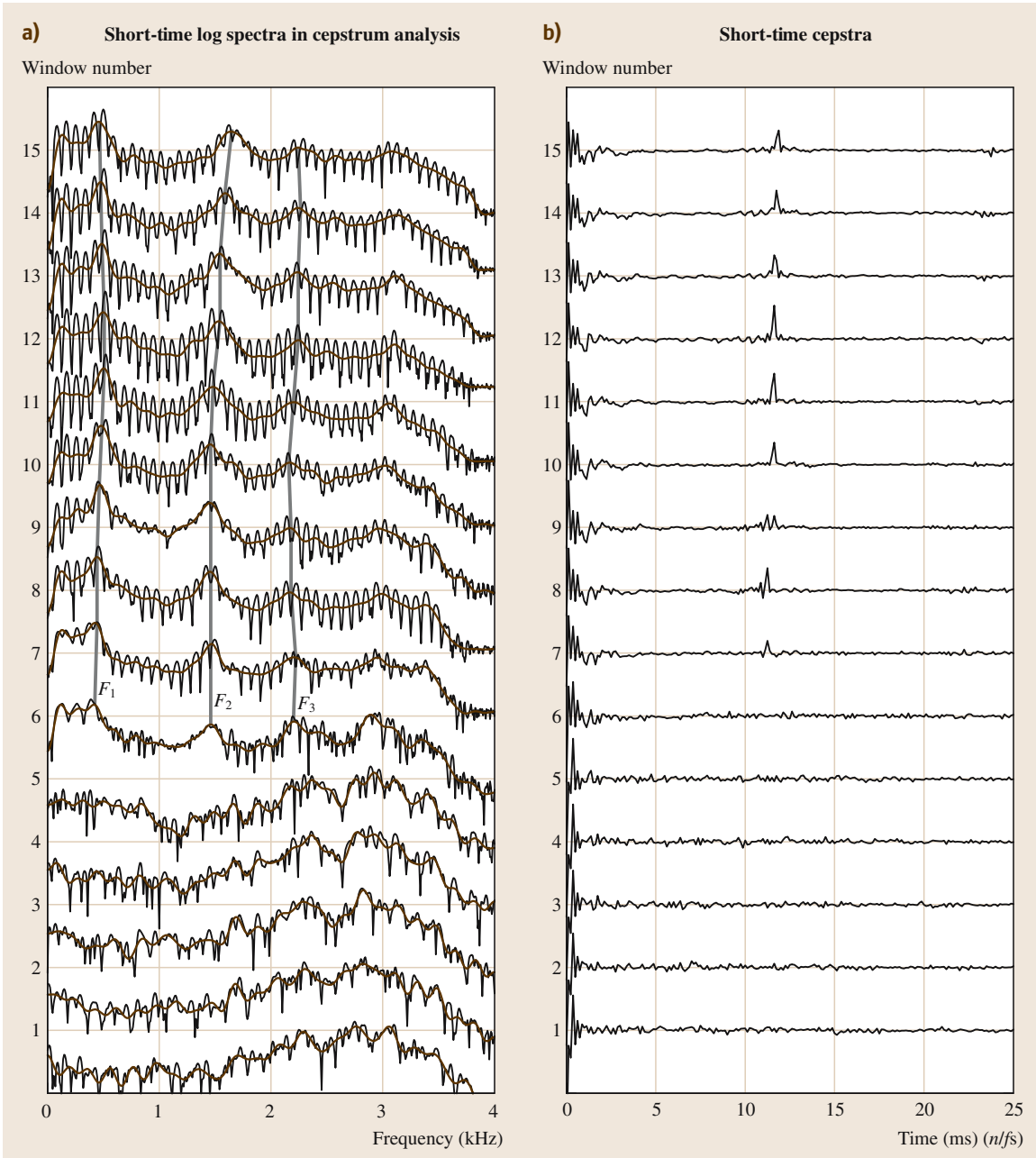


Fig. 9.13 (a) Short-time spectra and (b) cepstra of segment of speech in Fig. 9.6

9.7 Applications to Analysis/Synthesis Coding

In analysis/synthesis speech coding, the sampled speech signal is divided into frames (blocks of samples), and the excitation parameters and impulse response are determined for each successive frame. The cepstrum can

be used to estimate the parameters of the simple excitation model. For compression, the excitation parameters (voicing decision and pitch period) are quantized together with a quantized representation of the impulse response of the vocal-tract system. The composite quantized representation is then used for transmission or storage. To reconstitute the speech signal, the model of Fig. 9.7 is used with model parameters updated at the frame rate from the compressed data. For example, LPC vocoders use LPC analysis techniques to obtain the vocal-tract system model in the form of (9.18). This all-pole representation implies that the vocal-tract model can be implemented by a recursive digital filter for synthesis [9.12].

The cepstrum has also been used in analysis/synthesis speech coding to extract an estimate of the linear system in the speech model of Fig. 9.7. We shall discuss three approaches based on homomorphic methods.

9.7.1 Homomorphic Vocoder

A homomorphic vocoder is an analysis/synthesis speech coding system that uses a cepstrum pitch detector as described in Sect. 9.6 to estimate the excitation parameters and short-time homomorphic deconvolution to estimate the time varying impulse response of the speech model of Fig. 9.7 [9.13]. More specifically, the vocal-tract impulse response is extracted by homomorphic filtering from the same short-time cepstrum that is used for pitch detection. Each frame of the speech signal is represented by quantized versions of voiced/unvoiced decision, pitch period, gain, and a set of low-quefrency cepstrum values to represent the vocal tract impulse response. Since the cepstrum is an even function of n, only $c[n]$ for $n = 1, \ldots, n_{co}$ need be retained in order to construct $c[-n]$. Furthermore, we can assume that $c[0] = \log G$.

Synthesis is done by discrete convolution of an impulse response (reconstructed from the quantized low-quefrency cepstrum values) with an excitation signal constructed from the voicing, pitch period, and gain information. For voiced speech frames, the excitation is a train of unit impulses with spacing P_0 (as estimated from the cepstrum), while for unvoiced frames, the excitation is a discrete random noise sequence. The gain G is used to provide the correct signal amplitude after synthesis.

Since the cepstrum (not the complex cepstrum) is used to represent the linear filter for each frame, it is not possible to estimate the fully fledged mixed-phase impulse response as in the example of Fig. 9.8c. The cepstrum only represents the log magnitude of the short-time spectrum. Therefore, it is necessary to generate an appropriate phase to pair with the smoothed log magnitude function. This can be achieved as part of the liftering that extracts the low-quefrency values of the cepstrum. The simplest approach is to simply assign zero for the phase at each frequency. This is the case if we define

$$\hat{h}_z[n] = \ell_z[n]c[n] \,, \tag{9.39}$$

where

$$\ell_z[n] = \begin{cases} 1 & |n| \le n_{co} \\ 0 & \text{otherwise} \end{cases} \,. \tag{9.40}$$

This situation is depicted in Fig. 9.10b. Since $\hat{h}_z[n] = \hat{h}_z[-n]$, it follows that $\hat{H}_z(e^{i\omega})$ is purely real. Therefore, if $\hat{h}_z[n]$ is the input to the inverse characteristic system $D_*^{-1}\{\,\}$ (implemented with the DFT), then the resulting output $h_z[n]$ will have even symmetry as well. As an example, Fig. 9.14a shows the *zero-phase* impulse response that has the same log magnitude function as the mixed-phase impulse response of Fig. 9.8c, i. e., the smooth function in Fig. 9.9a .(The zero-phase impulse response has been truncated to the range $-80 \le n \le 80$

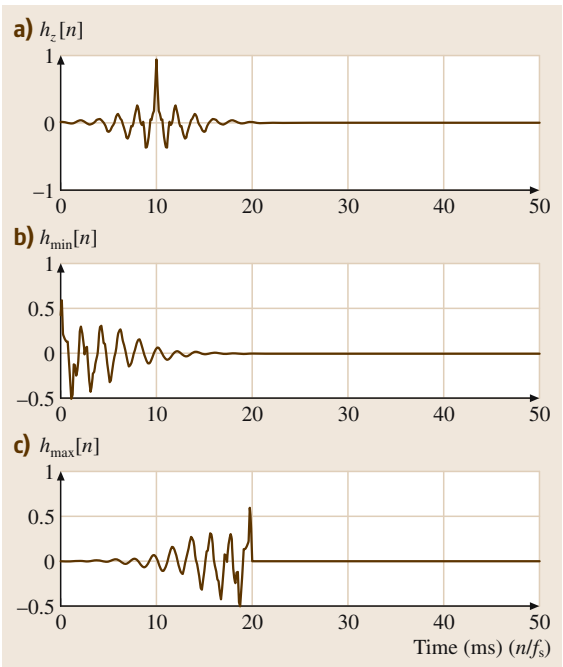


Fig. 9.14 **(a)** Minimum-phase impulse response. **(b)** Maximum-phase impulse response. **(c)** Zero-phase impulse response

and a delay of 80 samples has been included to make the impulse response causal.)

Another way of pairing a phase function with the smoothed log magnitude is to impose the minimum-phase condition, i. e., the complex cepstrum is zero for $n < 0$. Specifically, if $\hat{h}_{\min}[n] = \ell_{\min}[n]c[n]$, where

$$\ell_{\min}[n] = \begin{cases} 2 & 1 \le n \le n_{co} \\ 1 & n = 0 \\ 0 & \text{otherwise} \end{cases}, \tag{9.41}$$

then the resulting impulse response $h_{\min}[n]$ at the output of the inverse characteristic system is a minimum-phase impulse response whose Fourier transform has the same log magnitude as does $h_z[n]$ corresponding to (9.39), and whose phase is the minimum-phase function that would be obtained through the discrete *Hilbert* transform [9.4]. Figure 9.14b shows the minimum-phase impulse response that has the same Fourier transform log magnitude as the impulse responses in Figs. 9.8c and 9.14a.

Still another way to obtain a nonzero phase is to impose a *maximum-phase* condition; i. e., the complex cepstrum is zero for $n > 0$. The maximum-phase impulse response reconstructed from the cepstrum with $\ell_{\max}[n] = \ell_{\min}[-n]$ will be simply $h_{\max}[n] = h_{\min}[-n]$. For our example the maximum-phase impulse response is shown in Fig. 9.14c. [Since $h_{\max}[n] = h_{\min}[-n]$ is noncausal, we have included a delay of 160 samples (20 ms) to make it causal.]

The complex cepstrum is not generally used in the homomorphic vocoder because it would be necessary to quantize and encode the complex cepstrum values for $-n_{co} \le n \le n_{co}$ as opposed to just $0 \le n \le n_{co}$ for the zero-phase, minimum-phase, and maximum-phase representations. *Oppenheim* [9.13] reported on experiments that showed that synthetic speech output using minimum-phase impulse responses was preferred slightly over synthesis with zero-phase impulse responses. Both minimum- and zero-phase synthesis were preferred to maximum-phase impulse response synthesis, even though the maximum-phase impulse response has the same wave shape and energy distribution (except time-reversed) as the minimum-phase impulse response. Oppenheim speculated that this was because the minimum-phase approach most closely matches the short-time phase of the natural vocal-tract system.

9.7.2 Homomorphic Formant Vocoder

Homomorphic processing has also been proposed as the basis for a formant vocoder [9.14]. Here the approach was like the homomorphic vocoder of Sect. 9.7.1 in that the excitation parameters were derived from a cepstrum pitch detector, but instead of extracting an impulse response at each frame, the first three formant (resonance) frequencies were estimated and tracked in time from the liftered short-time log magnitude spectrum. These three (quantized) formant frequency estimates were used to represent the vocal tract filter for each analysis frame. For synthesis, the three formant frequencies were used to control the complex pole locations of three second-order sections of a 10th-order cascade-form recursive digital filter; a sampling rate of 8 kHz was used. The remaining four poles were fixed at values that give a natural overall average spectral shape.

The set of plots in the left panel of Fig. 9.13 illustrates how the formants were estimated and tracked in the short-time log spectrum. Note that the smoothed log spectra contain numerous local maxima (peaks). However, from a detailed study of the acoustics of speech production and from extensive analysis of the short-time spectrum of speech, it is well established that the formant frequencies are confined to certain ranges of frequency simply by physical constraints [9.8]. Furthermore, the low-frequency peak in the spectra of Fig. 9.13 can be attributed to the glottal pulse spectrum. Thus, by locating the peaks in the liftered log spectrum that lie in the range of approximately 200–3000 Hz, it is possible to identify and track from frame to frame the first three formant frequencies for voiced speech segments, as illustrated in Fig. 9.13. For unvoiced speech, a much simpler model was proposed consisting of one complex-conjugate pair of zeros and one complex-conjugate pair of poles. The pole and zero locations were estimated from the liftered log magnitude spectra so as to match the overall general shape of the unvoiced spectrum [9.8].

Although this representation of speech is quite simplified and therefore bound to suffer in terms of quality of the synthetic speech that can be reconstructed, its virtue is that the information rate for transmitting the pitch period, gain, and three formant frequencies can be very low. With rather straightforward coding techniques, it was found that the information rate could be as low as 600 bits/s without degradation of the quality that could be achieved when the parameters were unquantized. For many talkers, the synthetic speech was highly intelligible and of acceptable quality for communications. However, the simplicity and the many fixed parameters of the synthesis model limit the quality that can be achieved over a wide range of voices. This is also true of both the LPC vocoder and the homomorphic vocoder of Sect. 9.7.1 with simplified voiced/unvoiced

excitation modeling. Such approaches typically achieve a level of speech quality reproduction that is acceptable for some applications at bit rates around 2400 bps, but not nearly transparent to modeling distortions. Such representations are degraded by coarser quantization of the parameters, but quality cannot be improved by simply representing the parameters with greater and greater accuracy.

9.7.3 Analysis-by-Synthesis Vocoder

As increased computational power became readily available in decade of the 1980s, speech coding researchers turned their attention toward more-sophisticated techniques for modeling the excitation in LPC vocoders. Such techniques as multipulse [9.15], code excitation (CELP) [9.16], and self-excitation [9.17] led to greatly improved output quality at somewhat higher bit rates than the classical {V/UV, pitch, gain} representation used in the LPC and homomorphic vocoders. All these approaches were based on the idea of *analysis by synthesis*. Although most of the innovations in analysis-by-synthesis methods were originally created in the context of LPC modeling of the vocal-tract filter, *Chung* [9.18, 19] demonstrated that the benefits of the improved excitation modeling also applied to the homomorphic vocoder.

Figure 9.15 shows a general block diagram for an analysis-by-synthesis homomorphic vocoder. In LPC-based coders of this type, the vocal-tract filter is recursive. The homomorphic analysis-by-synthesis coder uses an finite impulse response (FIR) filter derived from the cepstrum as discussed above. As in other coders of this type, analysis is done frame-by-frame. The analysis-by-synthesis excitation modeling is based on finding the excitation sequence $u[n]$ that minimizes the perceptually weighted error between the speech signal samples $x[n]$ and the reconstructed samples $s[n] = h[n] * u[n]$. Over a frame of length L samples, the perceptually weighted error is

$$\begin{aligned} d_{\mathrm{pw}}[n] &= (x[n] - h[n] * u[n]) * g_{\mathrm{pw}}[n] \\ &= x_{\mathrm{pw}}[n] - h_{\mathrm{pw}}[n] * u[n] \end{aligned} \tag{9.42}$$

where $g_{\mathrm{pw}}[n]$ is the impulse response of the perceptual weighting filter and $h_{\mathrm{pw}}[n] = h[n] * g_{\mathrm{pw}}[n]$ is the perceptually-weighted vocal-tract impulse response. The computation of the perceptual weighting filter and weighted impulse response for the homomorphic analysis-by-synthesis coder will be discussed below.

Irrespective of how the vocal-tract impulse response is derived, the basic excitation analysis proceeds by assuming that the excitation signal is composed of a set of N (one or more) scaled component signals in the form

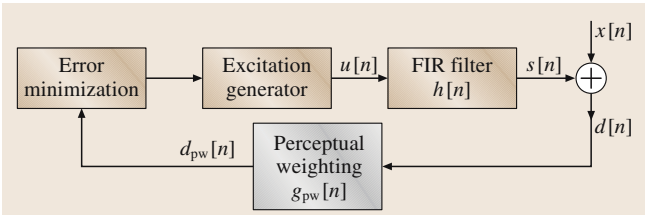


Fig. 9.15 Analysis-by-synthesis homomorphic vocoder

$$u[n] = \sum_{k=1}^{N} \beta_k u_{\gamma_k}[n] \,. \tag{9.43}$$

The set of excitation components $u_{\gamma_k}[n]$ is predetermined. In the case of multipulse [9.15], the components are shifted unit samples $\delta[n - \gamma_k]$; in CELP [9.16] the components are drawn from a library or codebook of white-noise sequences; and in the self-excited paradigm [9.17], the components are segments drawn from a memory of the past excitation sequence. In fact, the composite excitation in (9.43) can consist of a combination of such components [9.17].

The algorithm for determining $u[n]$ for a given frame begins by computing the output of the vocal-tract filter due to the excitation of previous frames. This is subtracted from the new frame of perceptually weighted speech to initialize the error. Then the analysis proceeds by a process something like the following to determine the k-th component:

1. Choose a particular component $u_\gamma[n]$ from the available set, and find the value of β that minimizes the total squared residual error over the frame.
2. Repeat step 1 for all possible input components. Choose the component (indexed by γ) that gave the lowest error, and subtract the output due to that component from the error.

These steps are repeated N times to obtain the indexes γ_k for $k = 1, 2, \ldots, N$. New values of the β_k can be determined after all the component sequences $u_{\gamma_k}[n]$ have been selected.

The main difference between LPC and homomorphic analysis-by-synthesis coders is the way in which the impulse responses are derived. In the case of the homomorphic vocoder, $h[n]$ is the quantized minimum-phase FIR vocal-tract impulse response obtained by low-pass liftering the cepstrum. The perceptual weighting filter $g_{\mathrm{pw}}[n]$ is derived directly from the cep-

strum according to the principles set out by *Atal* and *Schroeder* [9.20], who argued that the approximation errors in the regions of the formant peaks would be aurally masked, so the weighting should be designed to emphasize the regions between the formant peaks. In the homomorphic vocoder system, the perceptually weighted vocal tract impulse response is obtained directly from the cepstrum by [9.18]

$$\hat{h}_{\mathrm{pw}}[n] = \ell_{\mathrm{pw}}[n]c[n]\,, \tag{9.44a}$$

where

$$\ell_{\mathrm{pw}}[n] = \begin{cases} 1 - \alpha \sin\left(\frac{\pi n}{2n_{\mathrm{co}}}\right) & 0 \le n \le n_{\mathrm{co}} \\ 0 & \text{otherwise} \end{cases}\,. \tag{9.44b}$$

This low-pass liftering of the cepstrum creates a perceptually weighted vocal-tract filter where the formant resonances are flattened out as suggested by *Atal* and *Schroeder* [9.20]. To obtain the perceptual weighting filter needed to prefilter the input in (9.42), we simply observe that, since $h_{\mathrm{pw}}[n] = h[n] * g_{\mathrm{pw}}[n]$, it follows that the complex cepstrum of $g_{\mathrm{pw}}[n]$ is

$$\begin{aligned} \hat{g}_{\mathrm{pw}}[n] &= \hat{h}_{\mathrm{pw}}[n] - \hat{h}[n] \\ &= (\ell_{\mathrm{pw}}[n] - 1)\hat{h}[n]\,. \end{aligned} \tag{9.45}$$

Figure 9.16 illustrates the perceptual weighting filter for the segment of speech used to create the earlier examples. The light-gray solid curve is the smoothed short-time log spectrum corresponding to the vocal-tract filter impulse response $h[n]$. The dark solid curve is the perceptually weighted vocal-tract filter corresponding to $h_{\mathrm{pw}}[n]$, and the dashed curve corresponds to the perceptual weighting filter $g_{\mathrm{pw}}[n]$. Note how the perceptual weighting filter tends to de-emphasize the spectral regions around the formant frequencies.

Chung [9.19] performed subjective tests of the homomorphic analysis-by-synthesis coder, and compared it to LPC-based coders. In a comparison where the same analysis structure was used with both LPC and homomorphic vocal-tract filters, he found no significant difference. However, in comparison to the state-of-the-art Department of Defense standard CELP coder available at the time, both his LPC and homomorphic systems were significantly inferior. He attributed this to enhancements such as interpolation of vocal-tract filters and interpolated codebook search that were implemented in the the DoD standard system and not in his LPC and homomorphic coders. While the results of these subjective tests were inconclusive, it was demonstrated that the homomorphic vocoder could be significantly improved by use of analysis-by-synthesis excitation analysis, and that homomorphic processing afforded new flexibilities in implementing perceptual weighting of the coding error.

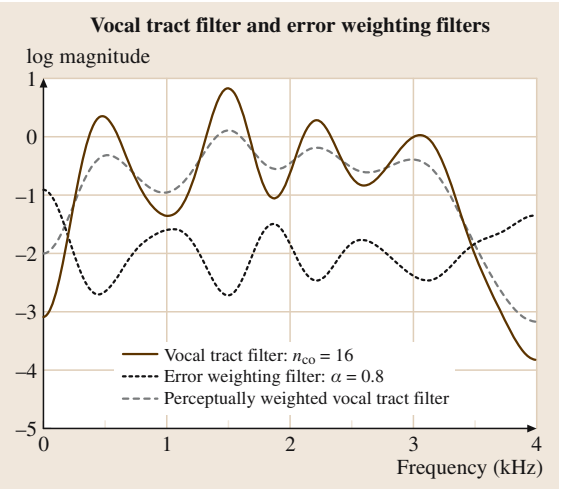


Fig. 9.16 Illustration of perceptual weighting filter derived from the cepstrum

9.8 Applications to Speech Pattern Recognition

Perhaps the most pervasive application of the cepstrum in speech processing is its use in pattern recognition problems such as vector quantization (VQ) and automatic speech recognition (ASR). In such applications, a speech signal is represented on a frame-by-frame basis by a sequence of short-time cepstra. In later discussions in this section, it will be useful to use somewhat more-complicated notation. Specifically, we denote the cepstrum of the m-th frame of a signal $x[n]$ as $c_m^{(x)}[n]$, where n denotes the quefrency index of the cepstrum. In cases where it is not necessary to distinguish between signals or frames, these additional designations will be omitted as we have done up to this point.

As we have shown, cepstra can be computed either by LPC analysis or by DFT implementation of the characteristic system. In either case, we can assume that the cepstrum vector corresponds to a gain-normalized ($c[0] = 0$) minimum-phase vocal-tract impulse response

that is defined by the complex cepstrum

$$\hat{h}[n] = \begin{cases} 2c[n] & 1 \le n \le n_{\mathrm{co}} \\ 0 & n < 0 \end{cases} , \tag{9.46}$$

where we have suppressed for now the notation $\overset{(x)}{m}$.

In problems such as VQ or ASR, a test pattern $c[n]$ (a vector of cepstrum values $n = 1, 2, \ldots, n_{\mathrm{co}}$) is compared against a comparable reference pattern $\bar{c}[n]$. Such comparisons require a distortion measure. For example, the Euclidean distance applied to the cepstrum would give

$$D = \sum_{n=1}^{n_{\mathrm{co}}} \left| c[n] - \bar{c}[n] \right|^2 . \tag{9.47a}$$

Equivalently in the frequency domain,

$$D = \frac{1}{4\pi} \int_{-\pi}^{\pi} \left| \log |H(\mathrm{e}^{\mathrm{i}\omega})| - \log |\bar{H}(\mathrm{e}^{\mathrm{i}\omega})| \right|^2 \mathrm{d}\omega , \tag{9.47b}$$

where $\log |H(\mathrm{e}^{\mathrm{i}\omega})|$ is the log magnitude of the DTFT of $h[n]$ corresponding to the complex cepstrum in (9.46) or the real part of the DTFT of $\hat{h}[n]$ in (9.46). Thus, cepstrum-based comparisons are strongly related to comparisons of smoothed short-time spectra.

The cepstrum offers an effective and flexible representation of speech for pattern recognition problems as we will discuss below.

9.8.1 Compensation for Linear Filtering

Suppose that we have only a linearly filtered version of the speech signal, $y[n] = h_d[n] * x[n]$, instead of $x[n]$. If the analysis window is long compared to the length of $h_d[n]$, the short-time cepstrum of one frame of the filtered speech signal $y[n]$ will be approximately

$$c_m^{(y)}[n] = c_m^{(x)}[n] + c^{(h_d)}[n] , \tag{9.48}$$

where $c^{(h_d)}[n]$ will appear more or less the same in each frame. Therefore, if we can estimate $c^{(h_d)}[n]$, which we assume is non-time-varying, we can obtain $c_m^{(x)}[n]$ at each frame from $c_y[n]$ by subtraction; i. e., $c_m^{(x)}[n] = c_m^{(y)}[n] - c^{(h_d)}[n]$. (*Stockham* [9.21] showed how $c^{(h_d)}[n]$ for such linear distortions can be estimated from the signal $y[n]$ by time averaging the log of the short-time Fourier transform.) This property is extremely attractive in situations where the set of reference patterns $\bar{c}[n]$ has been obtained under different recording or transmission conditions from those used to acquire the test vectors. In these circumstances, the test vectors can be compensated for the effects of the linear filtering prior to computing the distance measures used for comparison of patterns.

Another approach to removing the effects of linear distortions is to observe that the cepstrum component due to the distortion is the same in each frame. Therefore it can be removed by a simple first difference operation of the form

$$\Delta c_m^{(y)}[n] = c_m^{(y)}[n] - c_{m-1}^{(y)}[n] . \tag{9.49}$$

It is clear that, if $c_m^{(y)}[n] = c_m^{(x)}[n] + c^{(h_d)}[n]$ with $c^{(h_d)}[n]$ being independent of m, then $\Delta c_m^{(y)}[n] = \Delta c_m^{(x)}[n]$; i. e., the linear distortion effects are removed.

Furui [9.22] first noted that, in addition to making the cepstrum less susceptible to linear distortions, the sequence of cepstrum values has temporal information that could be of value for a speaker verification system. He used polynomial fits to cepstrum sequences to extract simple representations of the temporal variation. The *delta cepstrum* as defined in (9.49) is simply the slope of a first-order polynomial fit to the cepstrum time evolution.

9.8.2 Weighted Distance Measures

In using LPC analysis to obtain cepstrum feature vectors for pattern recognition problems, it was observed that there is significant statistical variability due to a variety of factors, including short-time analysis window position, bias toward harmonic peaks, and noise [9.23, 24]. A solution to this problem is to use weighted distance measures of the form

$$D = \sum_{n=1}^{n_{\mathrm{co}}} \ell^2[n] \left| c[n] - \bar{c}[n] \right|^2 , \tag{9.50a}$$

which can be written as the Euclidean distance of liftered cepstra

$$D = \sum_{n=1}^{n_{\mathrm{co}}} \left| \ell[n] c[n] - \ell[n] \bar{c}[n] \right|^2 . \tag{9.50b}$$

Tohkura [9.23] found, for example, that when averaged over many frames of speech and speakers, cepstrum values $c[n]$ have zero means and variances on the order of $1/n^2$. This suggests that $\ell[n] = n$ for $n = 1, 2, \ldots, n_{\mathrm{co}}$ could be used to equalize the contributions for each term to the cepstrum distance.

Juang et al. [9.24] observed that the variability due to the vagaries of LPC analysis could be lessened by using a high-pass lifter of the form

$$\ell[n] = 1 + 0.5 n_{\mathrm{co}} \sin(\pi n / n_{\mathrm{co}}) \,, \; n = 1, 2, \ldots, n_{\mathrm{co}} \tag{9.51}$$

Tests of weighted distance measures have shown consistent improvements in automatic speech recognition tasks [9.24].

9.8.3 Group Delay Spectrum

The weighted cepstral distance measures of the previous subsection were given a new interpretation by *Itakura* and *Umezaki* [9.25], who invoked the following basic property of the DTFT:

$$n\hat{h}[n] \Longleftrightarrow \mathrm{i}\frac{\mathrm{d}\hat{H}(\mathrm{e}^{\mathrm{i}\omega})}{\mathrm{d}\omega} \,, \tag{9.52}$$

where $\Longleftrightarrow$ denotes the unique relationship between a sequence and its DTFT. An interesting result can be obtained if we represent the complex cepstrum as

$$\hat{h}[n] = c[n] + g[n] \,, \tag{9.53}$$

where $c[n] = \mathrm{Ev}\{\hat{h}[n]\}$ is the even part and $g[n] = \mathrm{Odd}\{\hat{h}[n]\}$ is the odd part of the complex cepstrum. Recalling that the DTFT of the complex cepstrum is by definition $\hat{H}(\mathrm{e}^{\mathrm{i}\omega}) = \log|H(\mathrm{e}^{\mathrm{i}\omega})| + \mathrm{i}\arg\{H(\mathrm{e}^{\mathrm{i}\omega})\}$, it can be shown that the following DTFT relations hold:

$$nc[n] \Longleftrightarrow \mathrm{i}\frac{\mathrm{d}\log|H(\mathrm{e}^{\mathrm{i}\omega})|}{\mathrm{d}\omega} \tag{9.54a}$$

and

$$ng[n] \Longleftrightarrow -\frac{\mathrm{d}\arg\{H(\mathrm{e}^{\mathrm{i}\omega})\}}{\mathrm{d}\omega} \,. \tag{9.54b}$$

The DTFT expression on the right in (9.54b) is the group delay for the DTFT $H(\mathrm{e}^{\mathrm{i}\omega})$; i. e.,

$$\mathrm{grd}\{H(\mathrm{e}^{\mathrm{i}\omega})\} = -\frac{\mathrm{d}\arg\{H(\mathrm{e}^{\mathrm{i}\omega})\}}{\mathrm{d}\omega} \,. \tag{9.55}$$

Now if $h[n]$ is obtained by LPC analysis as discussed in Sect. 9.5, the complex cepstrum satisfies $\hat{h}[n] = 0$ for $n < 0$. This means that $\hat{h}[n] = 2c[n] = 2g[n]$ for $n > 0$. If we define $\ell[n] = n$, then

$$D = \sum_{n=-\infty}^{\infty} \left|\ell[n]g[n] - \ell[n]\bar{g}[n]\right| \tag{9.56a}$$

is equivalent to

$$D = \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|\mathrm{grd}\{H(\mathrm{e}^{\mathrm{i}\omega})\} - \mathrm{grd}\{\bar{H}(\mathrm{e}^{\mathrm{i}\omega})\}\right| \mathrm{d}\omega \tag{9.56b}$$

or, alternatively,

$$D = \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|\frac{\mathrm{d}\log|\bar{H}(\mathrm{e}^{\mathrm{i}\omega})|}{\mathrm{d}\omega} - \frac{\mathrm{d}\log|H(\mathrm{e}^{\mathrm{i}\omega})|}{\mathrm{d}\omega}\right| \mathrm{d}\omega \,. \tag{9.56c}$$

The result of (9.56c) was also given by *Tohkura* [9.23]. Instead of $\ell[n] = n$ for all n, or the lifter of (9.51), Itakura proposed the lifter

$$\ell[n] = n^s \mathrm{e}^{-n^2/2\tau^2} \,. \tag{9.57}$$

This lifter has great flexibility. For example, if $s = 0$ we have simply low-pass liftering of the cepstrum. If $s = 1$ and τ is large, we have essentially $\ell[n] = n$ for small n with high-quefrency tapering. The effect of liftering with (9.57) is illustrated in Fig. 9.17, which shows in (a) the short-time Fourier transform of a segment of voiced speech along with an LPC spectrum with $p = 12$. Figure 9.17b shows the liftered group delay spectrum for $s = 1$ and τ ranging from 5 to 35 in steps of 10. Observe that, as τ increases, the formant frequencies are increasingly emphasized. If larger values of s are used, even greater enhancement of the resonance structure is observed.

Itakura and *Umezaki* [9.25] tested the group delay spectrum distance measure in an automatic speech recognition system. They found that, for clean test utterances, the difference in recognition rate was small for

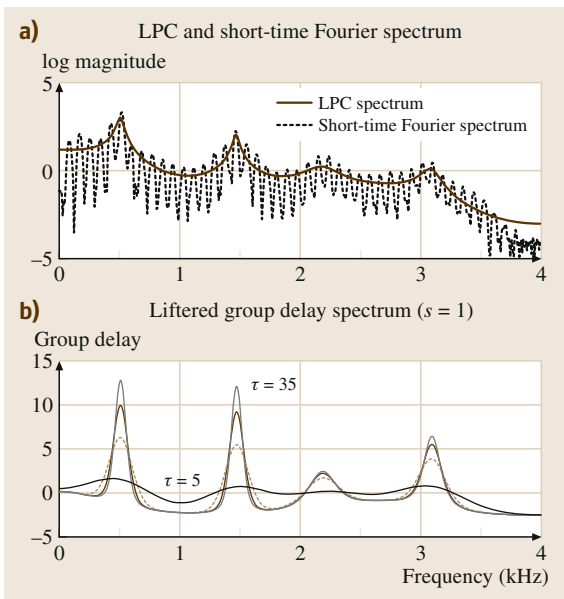


Fig. 9.17a,b Illustration of Itakura's liftered group delay spectrum (after [9.25])

different values of s when $\tau \approx 5$, although performance suffered with increasing s for larger values of τ. This was attributed to the fact that, for larger s, the group delay spectrum becomes very sharply peaked and thus more sensitive to small differences in formant locations. However, in test conditions with additive white noise and also with linear filtering distortions, recognition rates improved significantly with $\tau = 5$ and increasing values of the parameter s.

9.8.4 Mel-Frequency Cepstrum Coefficients (MFCC)

As we have seen, weighted cepstrum distance measures have a directly equivalent interpretation in terms of distance in the frequency domain. This is significant in light of models for the human perception of sound, which are based on a frequency analysis performed in the inner ear. With this in mind, *Davis* and *Mermelstein* [9.26] were motivated to formulate a new type of cepstrum representation that has come to be widely used and known as the mel-frequency cepstrum coefficients (MFCC).

The basic idea is to compute a frequency analysis based upon a filter bank with approximately critical band spacing of the filters and bandwidths. For 4 kHz bandwidth, approximately 20 filters are used. In most implementations, a short-time Fourier analysis is done first, resulting in a DFT $X_m[k]$ for the m-th frame. Then the DFT values are grouped together in critical bands and weighted by triangular weighting functions as depicted in Fig. 9.18. Note that the bandwidths in Fig. 9.18 are constant for center frequencies below 1 kHz and then increase exponentially up to half the sampling rate of 4 kHz, resulting in 24 *filters*. The mel-spectrum of the m-th frame is defined for $r = 1, 2, \ldots, R$ as

$$\mathrm{MF}_m[r] = \frac{1}{A_r} \sum_{k=L_r}^{U_r} |V_r[k] X_m[k]|^2 \qquad (9.58a)$$

where $V_r[k]$ is the weighting function for the r-th filter ranging from DFT index L_r to U_r, and

$$A_r = \frac{1}{A_r} \sum_{k=L_r}^{U_r} |V_r[k]|^2 \qquad (9.58b)$$

is a normalizing factor for the r-th mel-filter. This normalization is built into the weighting functions in Fig. 9.18. It is needed so that a perfectly flat input Fourier spectrum will produce a flat mel-spectrum. For each frame, a discrete cosine (DCT) transform of the log of the magnitude of the mel-filter outputs is computed to form the function mfcc$[n]$ as in

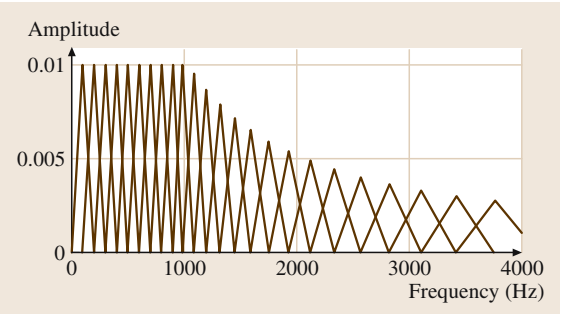


Fig. 9.18 Weighting functions for mel-scale filtering

$$\mathrm{mfcc}[n] = \frac{1}{R} \sum_{r=1}^{R} \log\left(\mathrm{MF}_m[r]\right) \cos\left[\frac{2\pi}{R}\left(r + \frac{1}{2}\right) n\right] \qquad (9.59)$$

Typically, mfcc$[n]$ is evaluated for a number of coefficients N_{mfcc} that is less than the number of mel-filters, e.g., $N_{\mathrm{mfcc}} = 13$ and $R = 24$. Figure 9.12 shows the result of MFCC analysis of a frame of voiced speech compared with the short-time spectrum, LPC spectrum, and two homomorphically smoothed spectra. The large dots are the values of $\log\left(\mathrm{MF}_m[r]\right)$ and the curve interpolated between them is a spectrum reconstructed at the original DFT frequencies. Note that all these spectra are different, but they have in common that they have peaks at the formant resonances. At higher frequencies, the reconstructed mel-spectrum of course has more smoothing due to the structure of the filter bank.

Note that the delta cepstrum idea expressed by (9.49) can be applied to MFCC to remove the effects of linear filtering as long as the frequency response of the distorting linear filter does not vary much across each of the mel-frequency bands.

The MFCC have become firmly established as the basic feature vector for many speech and acoustic pattern recognition problems. For this reason, new and efficient ways of computing MFCC are of interest. An intriguing proposal is to use floating gate electronic technology to implement the filter bank and the DCT computation with only microwatts of power [9.27].

9.9 Summary

This chapter has attempted to summarize the many ways in which the cepstrum has been used in speech processing. We started with definitions and showed many of the analytical results that flow from those definitions. These results have found application in many areas of speech and audio processing. With the advent of new approaches to cepstrum computation, new applications are likely to continue to emerge.

References

9.1 B.P. Bogert, M.J.R. Healy, J.W. Tukey: The quefrency alanysis of times series for echos: cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking, Proc. of the Symposium on Time Series Analysis, ed. by M. Rosenblatt (Wiley, New York 1963)

9.2 R.W. Schafer: *Echo removal by discrete generalized linear filtering* (MIT, Cambridge 1968), Ph.D. dissertation

9.3 A.V. Oppenheim, R.W. Schafer, T.G. Stockham Jr.: Nonlinear filtering of multiplied and convolved signals, Proc. IEEE **56**(8), 1264–1291 (1968)

9.4 A.V. Oppenheim, R.W. Schafer, J.R. Buck: *Discrete-Time Signal Processing* (Upper Saddle River, Prentice-Hall 1999)

9.5 A.V. Oppenheim: *Superposition in a Class of Nonlinear Systems* (MIT, Cambridge 1964), Ph.D. dissertation, Also: MIT Research Lab. of Electronics, Cambridge, Massachusetts, Technical Report 432

9.6 J.M. Tribolet: A new phase unwrapping algorithm, IEEE Trans. Acoust. Speech **ASSP-25**(2), 170–177 (1977)

9.7 G.A. Sitton, C.S. Burrus, J.W. Fox, S. Treitel: Factoring very-high-degree polynomials, IEEE Signal Proc. Mag. **20**(6), 27–42 (2003)

9.8 L.R. Rabiner, R.W. Schafer: *Digital Processing of Speech Signals* (Prentice-Hall, Englewood Cliffs 1978)

9.9 A.V. Oppenheim, R.W. Schafer: Homomorphic analysis of speech, IEEE Trans. Audio Electroacoust. **AU-16**, 221–228 (1968)

9.10 G.E. Kopec, A.V. Oppenheim, J.M. Tribolet: Speech analysis by homomorphic prediction, IEEE Trans. Acoust. Speech **ASSP-25**(1), 40–49 (1977)

9.11 A.M. Noll: Cepstrum pitch determination, J. Acoust. Soc. Am. **41**(2), 293–309 (1967)

9.12 B.S. Atal, S.L. Hanauer: Speech analysis and synthesis by linear prediction of the speech wave, J. Acoust. Soc. Am. **50**, 561–580 (1971)

9.13 A.V. Oppenheim: A speech analysis-synthesis system based on homomorphic filtering, J. Acoust. Soc. Am. **45**(2), 293–309 (1969)

9.14 R.W. Schafer, L.R. Rabiner: System for automatic formant analysis of voiced speech, J. Acoust. Soc. Am. **47**(2), 458–465 (1970)

9.15 B.S. Atal, J. Remde: A new model of LPC exitation for producing natural-sounding speech at low bit rates, Proc. IEEE ICASSP (1982), 614–617

9.16 M.R. Schroeder, B.S. Atal: Code-excited linear prediction (CELP): high-quality speech at very low bit rates, Proc. IEEE ICASSP (1985), 937–940

9.17 R.C. Rose, T.P. Barnwell III: The self excited vocoder – an alternate approach to toll quality at 4800 bps, Proc. IEEE ICASSP **11**, 453–456 (1986)

9.18 J.H. Chung, R.W. Schafer: Excitation modeling in a homomorphic vocoder, Proc. IEEE ICASSP **1**, 25–28 (1990)

9.19 J.H. Chung, R.W. Schafer: Performance evaluation of analysis-by-synthesis homomorphic vocoders, Proc. IEEE ICASSP **2**, 117–120 (1992)

9.20 B.S. Atal, M.R. Schroeder: Predictive coding of speech signals and subjective error criterion, IEEE Trans. Acoust. Speech **ASSP-27**, 247–254 (1079)

9.21 T.G. Stockham Jr., T.M. Cannon, R.B. Ingebretsen: Blind deconvolution through digital signal processing, Proc. IEEE **63**, 678–692 (1975)

9.22 S. Furui: Cepstral analysis technique for automatic speaker verification, IEEE Trans. Acoust. Speech **ASSP-29**(2), 254–272 (1981)

9.23 Y. Tohkura: A weighted cepstral distance measure for speech recognition, IEEE Trans. Acoust. Speech **ASSP-35**(10), 1414–1422 (1987)

9.24 B.-H. Juang, L.R. Rabiner, J.G. Wilpon: On the use of bandpass liftering in speech recognition, IEEE Trans. Acoust. Speech **ASSP-35**(7), 947–954 (1987)

9.25 F. Itakura, T. Umezaki: Distance measure for speech recognition based on the smoothed group delay spectrum, Proc. IEEE ICASSP **12**, 1257–1260 (1987)

9.26 S.B. Davis, P. Mermelstein: Comparison of parametric representations for monosyllabic word recognition in continously spoken sentences, IEEE Trans. Acoust. Speech **ASSP-28**(4), 357–366 (1980)

9.27 P.D. Smith, M. Kucic, R. Ellis, P. Hasler, D.V. Anderson: Mel-frequency cepstrum encoding in analog floating-gate circuitry, Proc. ISCAS **2002**(4), 671–674 (2002)

7. Linear Prediction

J. Benesty, J. Chen, Y. Huang

Linear prediction plays a fundamental role in all aspects of speech. Its use seems natural and obvious in this context since for a speech signal the value of its current sample can be well modeled as a linear combination of its past values. In this chapter, we attempt to present the most important ideas on linear prediction. We derive the principal results, widely recognized by speech experts, in a very intuitive way without sacrificing mathematical rigor.

7.1 Fundamentals

Linear prediction (LP) is a fundamental tool in many diverse areas such as adaptive filtering, system identification, economics, geophysics, spectral estimation, and speech. Recently, a nice history of LP in the context of speech coding was written by *Atal* [7.1]. Readers are invited to consult this reference for more information about this topic and how it has evolved.

Linear prediction is widely used in speech applications (recognition, compression, modeling, etc.) [7.2,3]. This is due to the fact that the speech production process is well modeled with LP. Indeed, it is well recognized that a speech signal can be written in the following form [7.4,5],

$$x(k) = \sum_{l=1}^{L} a_l x(k-l) + Gu(k)\,, \tag{7.1}$$

where k is the time index, L represents the number of coefficients in the model (the order of the predictor), a_l, $l = 1, \cdots, L$, are defined as the linear prediction coefficients, G is the gain of the system, and $u(k)$ is the excitation signal, which can be either a quasiperiodic train of impulses or a random noise source (also a combination of both signals for voiced fricatives such as 'v', 'z', and 'zh'). The periodic source produces voiced sounds such as vowels and nasals, and the noise source produces unvoiced or fricated sounds such as the fricatives. The parameters, a_l, determine the spectral characteristics of the particular sound for each of the two types of excitation and are widely used directly in many speech coding schemes and automatic speech recognition systems [7.4].

Equation (7.1) can be rewritten in the frequency domain, by using the z-transform. If $H(z)$ is the transfer function of the system, we have:

$$H(z) = \frac{G}{1 - \sum_{l=1}^{L} a_l z^{-l}} = \frac{G}{A(z)}\,, \tag{7.2}$$

which is an all-pole transfer function. This filter $[H(z)]$ is a good model of the human vocal tract [7.2]. Our main concern is to determine the predictor coefficients, $a_l, l = 1, 2, \cdots, L$, and to study the properties of the filter $A(z)$.

The applications of LP are numerous. Before addressing the estimation of LP coefficients, we give some examples to show the importance of LP. In many aspects of speech processing (noise reduction, speech separation, speech dereverberation, speech coding, etc.), it is of great interest to compare the closeness of the spectral envelope of two speech signals (the desired and

the processed ones) [7.6, 7]. One way of doing this is through comparing their LP coefficients. Consider the two speech signals $x(k)$ (desired) and $\hat{x}(k)$ (processed). Without entering too much into the details, one possible measure to evaluate the closeness of these two signals is the Itakura distance:

$$\mathrm{ID}_{x\hat{x}} = \ln \frac{E_x}{E_{\hat{x}}} , \tag{7.3}$$

where E_x and $E_{\hat{x}}$ are the prediction-error powers of the signals $x(k)$ and $\hat{x}(k)$, respectively (see the following sections for more details). Note that the Itakura distance is not symmetric, i. e.,

$$\mathrm{ID}_{x\hat{x}} \neq \mathrm{ID}_{\hat{x}x} , \tag{7.4}$$

therefore, it is not a distance metric. However, asymmetry is usually not a problem for applications such as speech quality evaluation.

A more-powerful distance was proposed by Itakura and Saito in their formulation of linear prediction as an approximate maximum-likelihood estimation [7.8]. This distance between the two signals $x(k)$ and $\hat{x}(k)$ is defined as,

$$\mathrm{ISD}_{x\hat{x}} = \frac{E_{\hat{x}}}{E_x} - \ln \frac{E_{\hat{x}}}{E_x} - 1 . \tag{7.5}$$

Like the Itakura distance, this measure is not symmetric either; therefore, it is not a true metric.

The Itakura–Saito distance has many interesting properties. It has been shown that this measure is highly correlated with subjective quality judgements [7.6]. For example, a recent report on speech codec evaluation reveals that, if the Itakura–Saito measure between two speech signals is less than 0.5, the difference in their mean opinion score would be less than 1.6 [7.9]. Many other reported experiments also confirmed that when the Itakura–Saito distance between two speech signals is below 0.1, they would be perceived nearly identically by human ears. As a result, the Itakura–Saito distance, which is based on LP, is often used as an objective measure of speech quality. It is probably the most widely used measure of similarity between speech signals.

The two previous examples of the vocal-tract filter and the speech quality measure clearly show the importance of LP in speech applications.

In this chapter, we study the theory of linear prediction and derive the most important LP techniques that are often encountered in many speech applications. We assume here that all signals of interest are real, stationary, and zero mean.

7.2 Forward Linear Prediction

Consider a stationary random signal $x(k)$. The objective of the forward linear prediction is to predict the value of the sample $x(k)$ from its past values, i. e., $x(k-1)$, $x(k-2)$, etc. We define the forward prediction error as [7.10, 11],

$$\begin{aligned} e_{\mathrm{f},L}(k) &= x(k) - \hat{x}(k) \\ &= x(k) - \sum_{l=1}^{L} a_{L,l}\, x(k-l) \\ &= x(k) - \boldsymbol{a}_L^{\mathrm{T}} \boldsymbol{x}(k-1) , \end{aligned} \tag{7.6}$$

where the superscript 'T' denotes transposition, $\hat{x}(k)$ is the predicted sample,

$$\boldsymbol{a}_L = [a_{L,1} \;\; a_{L,2} \;\; \cdots \;\; a_{L,L}]^{\mathrm{T}}$$

is the forward predictor of length L, and

$$\boldsymbol{x}(k-1) = [x(k-1) \;\; x(k-2) \;\; \cdots \;\; x(k-L)]^{\mathrm{T}}$$

is a vector containing the L most recent samples starting with and including $x(k-1)$.

We would like to find the optimal Wiener predictor. For that, we seek to minimize the mean-square error (MSE):

$$J_{\mathrm{f}}(\boldsymbol{a}_L) = E\left\{e_{\mathrm{f},L}^2(k)\right\} , \tag{7.7}$$

where $E\{\cdot\}$ denotes mathematical expectation. Taking the gradient of $J_{\mathrm{f}}(\boldsymbol{a}_L)$ with respect to $\boldsymbol{a}_L$ and equating to $\boldsymbol{0}_{L\times 1}$ (a vector of length L containing only zeroes), we easily find the Wiener–Hopf equations:

$$\mathbf{R}_L \boldsymbol{a}_{\mathrm{o},L} = \boldsymbol{r}_{\mathrm{f},L} , \tag{7.8}$$

where the subscript 'o' in $\boldsymbol{a}_{\mathrm{o},L}$ stands for optimal,

$$\begin{aligned} \mathbf{R}_L &= E\{\boldsymbol{x}(k-1)\boldsymbol{x}^{\mathrm{T}}(k-1)\} \\ &= E\{\boldsymbol{x}(k)\boldsymbol{x}^{\mathrm{T}}(k)\} \\ &= \begin{pmatrix} r(0) & r(1) & \cdots & r(L-1) \\ r(1) & r(0) & \cdots & r(L-2) \\ \vdots & \vdots & \ddots & \vdots \\ r(L-1) & r(L-2) & \cdots & r(0) \end{pmatrix} \end{aligned} \tag{7.9}$$

is the correlation matrix, and

$$\boldsymbol{r}_{\mathrm{f},L} = E\{\boldsymbol{x}(k-1)x(k)\} = [r(1)\ r(2)\ \cdots\ r(L)]^{\mathrm{T}} \tag{7.10}$$

is the correlation vector. The matrix $\mathbf{R}_L$ has a Toeplitz structure (i. e., all the entries along the diagonals are the same); assuming that it is nonsingular, we deduce the optimal forward predictor:

$$\boldsymbol{a}_{\mathrm{o},L} = \mathbf{R}_L^{-1}\boldsymbol{r}_{\mathrm{f},L}\ . \tag{7.11}$$

Expanding $e_{\mathrm{f},L}^2(k)$ in (7.7) and using (7.8) shows that the minimum mean-square error (MMSE),

$$J_{\mathrm{f,min}} = J_{\mathrm{f}}(\boldsymbol{a}_{\mathrm{o},L}) = r(0) - \boldsymbol{r}_{\mathrm{f},L}^{\mathrm{T}}\boldsymbol{a}_{\mathrm{o},L} = E_{\mathrm{f},L}\ . \tag{7.12}$$

This is also called the forward prediction-error power.

Define the augmented correlation matrix:

$$\mathbf{R}_{L+1} = \begin{pmatrix} r(0) & \boldsymbol{r}_{\mathrm{f},L}^{\mathrm{T}} \\ \boldsymbol{r}_{\mathrm{f},L} & \mathbf{R}_L \end{pmatrix}\ , \tag{7.13}$$

equations (7.8) and (7.12) may be combined in a convenient way:

$$\mathbf{R}_{L+1}\begin{pmatrix} 1 \\ -\boldsymbol{a}_{\mathrm{o},L} \end{pmatrix} = \begin{pmatrix} E_{\mathrm{f},L} \\ \boldsymbol{0}_{L\times 1} \end{pmatrix}\ . \tag{7.14}$$

We refer to (7.14) as the augmented Wiener–Hopf equations of a forward predictor of order L. From (7.13) we derive that,

$$\det(\mathbf{R}_{L+1}) = E_{\mathrm{f},L}\det(\mathbf{R}_L)\ , \tag{7.15}$$

where 'det' stands for determinant.

Let us now write the forward prediction errors for the optimal predictors of orders L and $L-i$:

$$e_{\mathrm{f,o},L}(k) = x(k) - \sum_{l=1}^{L} a_{\mathrm{o},L,l}\,x(k-l)\ , \tag{7.16}$$

$$e_{\mathrm{f,o},L-i}(k) = x(k) - \sum_{l=1}^{L-i} a_{\mathrm{o},L-i,l}\,x(k-l)\ . \tag{7.17}$$

From the principle of orthogonality [7.11], we know that:

$$E\{e_{\mathrm{f,o},L}(k)\boldsymbol{x}(k-1)\} = \boldsymbol{0}_{L\times 1}\ . \tag{7.18}$$

For $1 \le i \le L$, we can verify by using (7.18), that:

$$E\{e_{\mathrm{f,o},L}(k)e_{\mathrm{f,o},L-i}(k-i)\} = 0\ . \tag{7.19}$$

As a result,

$$\lim_{L\to\infty} E\{e_{\mathrm{f,o},L}(k)e_{\mathrm{f,o},L-i}(k-i)\} = E\{e_{\mathrm{f,o}}(k)e_{\mathrm{f,o}}(k-i)\} = 0\ . \tag{7.20}$$

This indicates that the signal $e_{\mathrm{f,o}}(k)$ is a white noise. So the optimal forward predictor has this important property of being able to whiten a stationary random process, provided that the order of the predictor is high enough.

7.3 Backward Linear Prediction

The aim of the backward linear prediction is to predict the value of the sample $x(k-L)$ from its future values, i. e., $x(k)$, $x(k-1)$, $\cdots$, $x(k-L+1)$. We define the backward prediction error as,

$$e_{\mathrm{b},L}(k) = x(k-L) - \hat{x}(k-L) = x(k-L) - \sum_{l=1}^{L} b_{L,l}\,x(k-l+1) = x(k-L) - \boldsymbol{b}_L^{\mathrm{T}}\boldsymbol{x}(k)\ , \tag{7.21}$$

where $\hat{x}(k-L)$ is the predicted sample,

$$\boldsymbol{b}_L = [b_{L,1}\ \ b_{L,2}\ \cdots\ b_{L,L}]^{\mathrm{T}}$$

is the backward predictor of order L, and

$$\boldsymbol{x}(k) = [x(k)\ \ x(k-1)\ \cdots\ x(k-L+1)]^{\mathrm{T}}\ .$$

The minimization of the MSE,

$$J_{\mathrm{b}}(\boldsymbol{b}_L) = E\left\{e_{\mathrm{b},L}^2(k)\right\}\ , \tag{7.22}$$

leads to the Wiener–Hopf equations:

$$\mathbf{R}_L\boldsymbol{b}_{\mathrm{o},L} = \boldsymbol{r}_{\mathrm{b},L}\ , \tag{7.23}$$

where

$$\boldsymbol{r}_{\mathrm{b},L} = E\{\boldsymbol{x}(k)x(k-L)\} = [r(L)\ \ r(L-1)\ \cdots\ r(1)]^{\mathrm{T}}\ . \tag{7.24}$$

Therefore, the optimal backward predictor is:

$$\boldsymbol{b}_{\mathrm{o},L} = \mathbf{R}_L^{-1}\boldsymbol{r}_{\mathrm{b},L}\ . \tag{7.25}$$

The MMSE for backward prediction,

$$J_{\mathrm{b,min}} = J_{\mathrm{b}}(\boldsymbol{b}_{\mathrm{o},L}) = r(0) - \boldsymbol{r}_{\mathrm{b},L}^{\mathrm{T}}\boldsymbol{b}_{\mathrm{o},L} = E_{\mathrm{b},L}\ , \tag{7.26}$$

is also called the backward prediction-error power.

Define the augmented correlation matrix:

$$\mathbf{R}_{L+1} = \begin{pmatrix} \mathbf{R}_L & \boldsymbol{r}_{\mathrm{b},L} \\ \boldsymbol{r}_{\mathrm{b},L}^{\mathrm{T}} & r(0) \end{pmatrix}\ , \tag{7.27}$$

equations (7.23) and (7.26) may be combined in a convenient way:

$$\mathbf{R}_{L+1}\begin{pmatrix}-\boldsymbol{b}_{\mathrm{o},L}\\1\end{pmatrix}=\begin{pmatrix}\boldsymbol{0}_{L\times 1}\\E_{\mathrm{b},L}\end{pmatrix}\,. \tag{7.28}$$

We refer to this expression as the augmented Wiener–Hopf equations of a backward predictor of order L.

One important property of backward prediction is that the error signals of different orders with the optimal predictors are uncorrelated, i. e., $E\{e_{\mathrm{b,o},i}(k)e_{\mathrm{b,o},l}(k)\}=0, i\neq l, i,l=0,1,\cdots,L-1$. To prove this, let us rewrite the error signal in vector form:

$$\boldsymbol{e}_{\mathrm{b,o}}(k)=\mathbf{L}\boldsymbol{x}(k)\,, \tag{7.29}$$

where

$$\boldsymbol{e}_{\mathrm{b,o}}(k)=[e_{\mathrm{b,o},0}(k)\;\; e_{\mathrm{b,o},1}(k)\;\;\cdots\;\; e_{\mathrm{b,o},L-1}(k)]^{\mathrm{T}} \tag{7.30}$$

and

$$\mathbf{L}=\begin{pmatrix}1 & 0 & 0 & \cdots & 0\\ -\boldsymbol{b}_{\mathrm{o},1}^{\mathrm{T}} & 1 & 0 & \cdots & 0\\ -\boldsymbol{b}_{\mathrm{o},2}^{\mathrm{T}} & & 1 & \cdots & 0\\ \vdots & \vdots & & \vdots & \vdots\\ -\boldsymbol{b}_{\mathrm{o},L-1}^{\mathrm{T}} & & & & 1\end{pmatrix} \tag{7.31}$$

is a lower triangular matrix with 1s along its main diagonal. The covariance matrix corresponding to the vector signal $\boldsymbol{e}_{\mathrm{b,o}}(k)$ is:

$$E\left\{\boldsymbol{e}_{\mathrm{b,o}}(k)\boldsymbol{e}_{\mathrm{b,o}}^{\mathrm{T}}(k)\right\}=\mathbf{L}\mathbf{R}_L\mathbf{L}^{\mathrm{T}}\,. \tag{7.32}$$

By definition, the previous matrix is symmetric. The matrix product $\mathbf{R}_L\mathbf{L}^{\mathrm{T}}$ is a lower triangular matrix because of (7.28) and the main diagonal contains the backward prediction-error powers $E_{\mathrm{b},l}$ $(0\leq l\leq L-1)$. Since $\mathbf{L}$ is also a lower triangular matrix, the product between the two matrices $\mathbf{L}$ and $\mathbf{R}_L\mathbf{L}^{\mathrm{T}}$ should have the same structure and, since it has to be symmetric, the only possibility is that this resulting matrix is diagonal:

$$E\left\{\boldsymbol{e}_{\mathrm{b,o}}(k)\boldsymbol{e}_{\mathrm{b,o}}^{\mathrm{T}}(k)\right\}=\mathrm{diag}[E_{\mathrm{b},0},E_{\mathrm{b},1},\cdots,E_{\mathrm{b},L-1}]\,, \tag{7.33}$$

and hence the prediction errors are uncorrelated.

Furthermore,

$$\mathbf{L}\mathbf{R}_L\mathbf{L}^{\mathrm{T}}=\mathrm{diag}[E_{\mathrm{b},0},E_{\mathrm{b},1},\cdots,E_{\mathrm{b},L-1}]\,, \tag{7.34}$$

taking the inverse of the previous equation,

$$\mathbf{L}^{-\mathrm{T}}\mathbf{R}_L^{-1}\mathbf{L}^{-1}=\mathrm{diag}\left[E_{\mathrm{b},0}^{-1},E_{\mathrm{b},1}^{-1},\cdots,E_{\mathrm{b},L-1}^{-1}\right]\,, \tag{7.35}$$

we finally get:

$$\mathbf{R}_L^{-1}=\mathbf{L}^{\mathrm{T}}\mathrm{diag}\left[E_{\mathrm{b},0}^{-1},E_{\mathrm{b},1}^{-1},\cdots,E_{\mathrm{b},L-1}^{-1}\right]\mathbf{L}\,. \tag{7.36}$$

Expression (7.36) defines the Cholesky factorization of the inverse matrix $\mathbf{R}_L^{-1}$ [7.10, 12].

7.4 Levinson–Durbin Algorithm

The Levinson–Durbin algorithm is an efficient way to solve the Wiener–Hopf equations for the forward and backward prediction coefficients. This efficient method can be derived thanks to the Toeplitz structure of the correlation matrix $\mathbf{R}_L$. This algorithm was first invented by *Levinson* [7.13] and independently reformulated at a later date by *Durbin* [7.14, 15]. *Burg* gave a more-elegant presentation [7.16]. Before describing this algorithm, we first need to show some important relations between the forward and backward predictors.

We define the co-identity matrix as:

$$\mathbf{J}_L=\begin{pmatrix}0 & 0 & \cdots & 0 & 1\\ 0 & 0 & \cdots & 1 & 0\\ \vdots & \vdots & \ddots & \vdots & \vdots\\ 0 & 1 & \cdots & 0 & 0\\ 1 & 0 & \cdots & 0 & 0\end{pmatrix}\,.$$

We can easily check that:

$$\mathbf{R}_L\mathbf{J}_L=\mathbf{J}_L\mathbf{R}_L\,. \tag{7.37}$$

The matrix $\mathbf{R}_L$ is said to be persymmetric. We also have, $\boldsymbol{r}_{\mathrm{f},L}=\mathbf{J}_L\boldsymbol{r}_{\mathrm{b},L}$. If we left-multiply both sides of the Wiener–Hopf equations (7.23) by $\mathbf{J}_L$, we get:

$$\begin{aligned}\mathbf{J}_L\mathbf{R}_L\boldsymbol{b}_{\mathrm{o},L}&=\mathbf{J}_L\boldsymbol{r}_{\mathrm{b},L}=\boldsymbol{r}_{\mathrm{f},L}\\&=\mathbf{R}_L\mathbf{J}_L\boldsymbol{b}_{\mathrm{o},L}=\mathbf{R}_L\boldsymbol{a}_{\mathrm{o},L}\,,\end{aligned} \tag{7.38}$$

and, assuming that $\mathbf{R}_L$ is nonsingular, we see that:

$$\boldsymbol{a}_{\mathrm{o},L}=\mathbf{J}_L\boldsymbol{b}_{\mathrm{o},L}\,. \tag{7.39}$$

Furthermore,

$$\begin{aligned} E_{\mathrm{b},L} &= r(0) - \boldsymbol{r}_{\mathrm{b},L}^{\mathrm{T}} \boldsymbol{b}_{\mathrm{o},L} \\ &= r(0) - \boldsymbol{r}_{\mathrm{b},L}^{\mathrm{T}} \mathbf{J}_L \mathbf{J}_L \boldsymbol{b}_{\mathrm{o},L} \\ &= r(0) - \boldsymbol{r}_{\mathrm{f},L}^{\mathrm{T}} \boldsymbol{a}_{\mathrm{o},L} \\ &= E_{\mathrm{f},L} = E_L \,. \end{aligned} \tag{7.40}$$

Therefore, for a stationary process, the forward and backward prediction-error powers are equal and the coefficients of the optimal forward predictor are the same as those of the optimal backward predictor, but in a reverse order.

The Levinson–Durbin algorithm is based on recursions of the orders of the prediction equations. Consider the following expression,

$$\begin{pmatrix} \mathbf{R}_L & \boldsymbol{r}_{\mathrm{b},L} \\ \boldsymbol{r}_{\mathrm{b},L}^{\mathrm{T}} & r(0) \end{pmatrix} \begin{pmatrix} 1 \\ -\boldsymbol{a}_{\mathrm{o},L-1} \\ 0 \end{pmatrix} = \begin{pmatrix} E_{L-1} \\ \boldsymbol{0}_{(L-1)\times 1} \\ K_L \end{pmatrix} , \tag{7.41}$$

where

$$\begin{aligned} K_L &= r(L) - \boldsymbol{a}_{\mathrm{o},L-1}^{\mathrm{T}} \boldsymbol{r}_{\mathrm{b},L-1} \\ &= r(L) - \boldsymbol{a}_{\mathrm{o},L-1}^{\mathrm{T}} \mathbf{J}_{L-1} \boldsymbol{r}_{\mathrm{f},L-1} \,. \end{aligned} \tag{7.42}$$

We define the reflection coefficient as,

$$\kappa_L = \frac{K_L}{E_{L-1}} \,. \tag{7.43}$$

From backward linear prediction, we have:

$$\begin{pmatrix} r(0) & \boldsymbol{r}_{\mathrm{f},L}^{\mathrm{T}} \\ \boldsymbol{r}_{\mathrm{f},L} & \mathbf{R}_L \end{pmatrix} \begin{pmatrix} 0 \\ -\boldsymbol{b}_{\mathrm{o},L-1} \\ 1 \end{pmatrix} = \begin{pmatrix} K_L \\ \boldsymbol{0}_{(L-1)\times 1} \\ E_{L-1} \end{pmatrix} \,. \tag{7.44}$$

Multiplying both sides of the previous equation by κ_L, we get,

$$\mathbf{R}_{L+1} \begin{pmatrix} 0 \\ -\kappa_L \boldsymbol{b}_{\mathrm{o},L-1} \\ \kappa_L \end{pmatrix} = \begin{pmatrix} \kappa_L^2 E_{L-1} \\ \boldsymbol{0}_{(L-1)\times 1} \\ K_L \end{pmatrix} \,. \tag{7.45}$$

If we now subtract (7.45) from (7.41), we obtain,

$$\mathbf{R}_{L+1} \begin{pmatrix} 1 \\ \kappa_L \boldsymbol{b}_{\mathrm{o},L-1} - \boldsymbol{a}_{\mathrm{o},L-1} \\ -\kappa_L \end{pmatrix} = \begin{pmatrix} E_{L-1}(1-\kappa_L^2) \\ \boldsymbol{0}_{L\times 1} \end{pmatrix} \,. \tag{7.46}$$

Assuming that $\mathbf{R}_{L+1}$ is nonsingular and identifying (7.46) with (7.14), we can deduce the recursive equations:

$$\boldsymbol{a}_{\mathrm{o},L} = \begin{pmatrix} \boldsymbol{a}_{\mathrm{o},L-1} \\ 0 \end{pmatrix} - \kappa_L \begin{pmatrix} \boldsymbol{b}_{\mathrm{o},L-1} \\ -1 \end{pmatrix} \,, \tag{7.47}$$

$$E_L = E_{L-1}(1-\kappa_L^2) \,, \tag{7.48}$$

$$a_{\mathrm{o},L,L} = \kappa_L \,. \tag{7.49}$$

Iterating on the prediction-error power given in (7.48), we find that,

$$E_L = r(0) \prod_{l=1}^{L} (1-\kappa_l^2) \,, \tag{7.50}$$

and since $E_L \geq 0$, this implies that,

$$|\kappa_l| \leq 1, \quad \forall l \geq 1 \,. \tag{7.51}$$

Also, from (7.48) we see that we have,

$$0 \leq E_l \leq E_{l-1}, \quad \forall l \geq 1 \,, \tag{7.52}$$

so, as the order of the predictors increases, the prediction-error power decreases.

Table 7.1 summarizes the Levinson–Durbin algorithm, whose arithmetic complexity is proportional to L^2. This algorithm is much more efficient than standard methods such as the Gauss elimination technique, whose complexity is on the order of L^3. The saving in number of operations to find the optimal Wiener predictor can be very important, especially when L is large. The other advantage of the Levinson–Durbin algorithm is that it gives the predictors of all orders and the algorithm can be stopped if the prediction-error power is under a threshold, which can be very useful in practice when the choice of the predictor order is not easy to get in advance. A slightly more-efficient approach, called the split Levinson algorithm, can be found in [7.17]. This algorithm requires roughly half the number of multiplications and the same number of additions as the classical Levison–Durbin algorithm. Even more-efficient algorithms have been proposed (see, for example, [7.18]) but they are numerically unstable, which is not acceptable in most speech applications.

Table 7.1 Levinson–Durbin algorithm

Initialization: $E_0 = r(0)$

For $1 \leq l \leq L$

$$\kappa_l = \frac{1}{E_{l-1}} \left[r(l) - \boldsymbol{a}_{\mathrm{o},l-1}^{\mathrm{T}} \mathbf{J}_{l-1} \boldsymbol{r}_{\mathrm{f},l-1} \right]$$

$$\boldsymbol{a}_{\mathrm{o},l} = \begin{pmatrix} \boldsymbol{a}_{\mathrm{o},l-1} \\ 0 \end{pmatrix} - \kappa_l \mathbf{J}_l \begin{pmatrix} -1 \\ \boldsymbol{a}_{\mathrm{o},l-1} \end{pmatrix}$$

$$E_l = E_{l-1}\left(1-\kappa_l^2\right)$$

7.5 Lattice Predictor

In this section, we will show that the order-recursive structure of the forward and backward prediction errors has the form of a ladder, which is called a lattice predictor.

Inserting (7.47) into the forward prediction error for the optimal predictor of order L,

$$e_{\mathrm{f,o},L}(k) = x(k) - \boldsymbol{a}_{\mathrm{o},L}^{\mathrm{T}} \boldsymbol{x}(k-1)\,, \tag{7.53}$$

we obtain,

$$\begin{aligned} e_{\mathrm{f,o},L}(k) &= e_{\mathrm{f,o},L-1}(k) \\ &\quad - \kappa_L \begin{pmatrix} -\boldsymbol{b}_{\mathrm{o},L-1}^{\mathrm{T}} & 1 \end{pmatrix} \boldsymbol{x}(k-1)\,. \end{aligned} \tag{7.54}$$

The second term (without the reflection coefficient) on the right-hand side of (7.54) is the backward prediction error, at time $k-1$, for the optimal predictor of order $L-1$. Therefore, (7.54) can be rewritten

$$e_{\mathrm{f,o},L}(k) = e_{\mathrm{f,o},L-1}(k) - \kappa_L e_{\mathrm{b,o},L-1}(k-1)\,. \tag{7.55}$$

If we insert (7.47) again into the backward prediction error for the optimal predictor of order L,

$$\begin{aligned} e_{\mathrm{b,o},L}(k) &= x(k-L) - \boldsymbol{b}_{\mathrm{o},L}^{\mathrm{T}} \boldsymbol{x}(k) \\ &= x(k-L) - \boldsymbol{a}_{\mathrm{o},L}^{\mathrm{T}} \boldsymbol{J}_L \boldsymbol{x}(k)\,, \end{aligned} \tag{7.56}$$

we get,

$$e_{\mathrm{b,o},L}(k) = e_{\mathrm{b,o},L-1}(k-1) - \kappa_L e_{\mathrm{f,o},L-1}(k)\,. \tag{7.57}$$

If we put (7.55) and (7.57) into a matrix form, we have,

$$\begin{aligned} \begin{pmatrix} e_{\mathrm{f,o},L}(k) \\ e_{\mathrm{b,o},L}(k) \end{pmatrix} &= \begin{pmatrix} 1 & -\kappa_L \\ -\kappa_L & 1 \end{pmatrix} \begin{pmatrix} e_{\mathrm{f,o},L-1}(k) \\ e_{\mathrm{b,o},L-1}(k-1) \end{pmatrix} \\ &= \prod_{l=1}^{L} \begin{pmatrix} 1 & -\kappa_l \\ -\kappa_l & 1 \end{pmatrix} \begin{pmatrix} x(k) \\ x(k-1) \end{pmatrix}\,, \end{aligned} \tag{7.58}$$

where we have taken for initial conditions (order 0), $e_{\mathrm{f,o},0}(k) = x(k)$ and $e_{\mathrm{b,o},0}(k-1) = x(k-1)$. Figure 7.1 depicts the l-th stage of a lattice predictor. For the whole lattice predictor, L of these stages are needed and are connected in cascade, one to each other, starting from order 0 to order L.

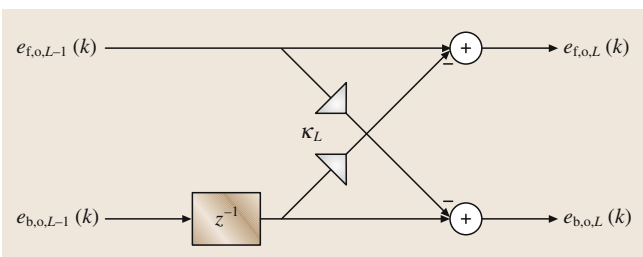


Fig. 7.1 Stage l of a lattice predictor

Now let us compute the variance of $e_{\mathrm{b,o},L}(k)$ from (7.57),

$$\begin{aligned} E\{e_{\mathrm{b,o},L}^2(k)\} &= E_L \\ &= E_{L-1} + \kappa_L^2 E_{L-1} \\ &\quad - 2\kappa_L E\{e_{\mathrm{f,o},L-1}(k) e_{\mathrm{b,o},L-1}(k-1)\} \\ &= E_{L-1}\left(1-\kappa_L^2\right)\,. \end{aligned} \tag{7.59}$$

Developing the previous expression, we obtain,

$$\begin{aligned} \kappa_L &= \frac{E\{e_{\mathrm{f,o},L-1}(k) e_{\mathrm{b,o},L-1}(k-1)\}}{E_{L-1}} \\ &= \frac{E\{e_{\mathrm{f,o},L-1}(k) e_{\mathrm{b,o},L-1}(k-1)\}}{\sqrt{E\{e_{\mathrm{f,o},L-1}^2(k)\} E\{e_{\mathrm{b,o},L-1}^2(k-1)\}}}\,. \end{aligned} \tag{7.60}$$

We see from (7.60) that the reflection coefficients are also the normalized cross-correlation coefficients between the forward and backward prediction errors, which is why they are also often called partial correlation (PARCOR) coefficients [7.3, 19, 20]. These coefficients are linked to the zeroes of the forward prediction-error FIR filter of order L, whose transfer function is

$$A_{\mathrm{o},L}(z) = 1 - \sum_{l=1}^{L} a_{\mathrm{o},L,l} z^{-l} = \prod_{l=1}^{L} \left(1 - z_{\mathrm{o},l} z^{-1}\right)\,, \tag{7.61}$$

where $z_{\mathrm{o},l}$ are the roots of $A_{\mathrm{o},L}(z)$. Since $\kappa_L = a_{\mathrm{o},L,L}$, we have,

$$\kappa_L = (-1)^{L+1} \prod_{l=1}^{L} z_{\mathrm{o},l}\,. \tag{7.62}$$

The filter $A_{\mathrm{o},L}(z)$ can be shown to be minimum phase, i. e., $|z_{\mathrm{o},l}| \le 1$, $\forall l$. As a result [because of the relation (7.39)], the filter $B_{\mathrm{o},L}(z)$ corresponding to the backward predictor is maximum phase. We will now show this very important property that the forward predictor is minimum phase. As far as we know, this simple and elegant proof was first shown by M. Mohan Sondhi but was never been published. A similar proof can be found in [7.21] and [7.22].

To avoid cumbersome notation, redefine the coefficients $w_l = -a_{\mathrm{o},L,l}$, with $w_0 = 1$, so that the polynomial

becomes,

$$A_{o,L}(z) = \sum_{l=0}^{L} w_l z^{-l} . \tag{7.63}$$

Also, define the vector,

$$\boldsymbol{w} = [w_0 \;\; w_1 \;\; \cdots \;\; w_L]^{\mathrm{T}} .$$

We know that,

$$\mathbf{R}_{L+1}\boldsymbol{w} = \begin{pmatrix} E_L \\ \boldsymbol{0}_{L\times 1} \end{pmatrix} . \tag{7.64}$$

If λ is a root of the polynomial, it follows that,

$$A_{o,L}(z) = (1-\lambda z^{-1}) \sum_{l=0}^{L-1} g_l z^{-l}, \text{ with } g_0 = 1 . \tag{7.65}$$

(Note that since λ can be complex, the coefficients g_l are, in general, complex.) Thus the vector $\boldsymbol{w}$ can be written

$$\boldsymbol{w} = \boldsymbol{g} - \lambda\tilde{\boldsymbol{g}} , \tag{7.66}$$

where

$$\boldsymbol{g} = [1 \;\; g_1 \;\; g_2 \;\; \cdots \;\; g_{L-1} \;\; 0]^{\mathrm{T}} = [\boldsymbol{g}'^{\mathrm{T}} \;\; 0]^{\mathrm{T}} ,$$
$$\tilde{\boldsymbol{g}} = [0 \;\; 1 \;\; g_1 \;\; g_2 \;\; \cdots \;\; g_{L-1}]^{\mathrm{T}} = [0 \;\; \boldsymbol{g}'^{\mathrm{T}}]^{\mathrm{T}} .$$

Substituting (7.66) in (7.64), we obtain,

$$\mathbf{R}_{L+1}\boldsymbol{g} = \lambda\mathbf{R}_{L+1}\tilde{\boldsymbol{g}} + \begin{pmatrix} E_L \\ \boldsymbol{0}_{L\times 1} \end{pmatrix} . \tag{7.67}$$

Now, premultiplying by $\tilde{\boldsymbol{g}}^{\mathrm{H}}$ (where the superscript $^{\mathrm{H}}$ denotes conjugate transpose) gives,

$$\tilde{\boldsymbol{g}}^{\mathrm{H}}\mathbf{R}_{L+1}\boldsymbol{g} = \lambda\tilde{\boldsymbol{g}}^{\mathrm{H}}\mathbf{R}_{L+1}\tilde{\boldsymbol{g}} . \tag{7.68}$$

Thus,

$$\left|\tilde{\boldsymbol{g}}^{\mathrm{H}}\mathbf{R}_{L+1}\boldsymbol{g}\right|^2 = |\lambda|^2 \left(\tilde{\boldsymbol{g}}^{\mathrm{H}}\mathbf{R}_{L+1}\tilde{\boldsymbol{g}}\right)^2 . \tag{7.69}$$

Using the Schwartz inequality,

$$\left|\tilde{\boldsymbol{g}}^{\mathrm{H}}\mathbf{R}_{L+1}\boldsymbol{g}\right|^2 \le \left(\tilde{\boldsymbol{g}}^{\mathrm{H}}\mathbf{R}_{L+1}\tilde{\boldsymbol{g}}\right)\left(\boldsymbol{g}^{\mathrm{H}}\mathbf{R}_{L+1}\boldsymbol{g}\right) . \tag{7.70}$$

However,

$$\begin{aligned}\tilde{\boldsymbol{g}}^{\mathrm{H}}\mathbf{R}_{L+1}\tilde{\boldsymbol{g}} &= \begin{pmatrix} 0 & \boldsymbol{g}'^{\mathrm{H}} \end{pmatrix} \begin{pmatrix} r(0) & \boldsymbol{r}_{\mathrm{f},L}^{\mathrm{T}} \\ \boldsymbol{r}_{\mathrm{f},L} & \mathbf{R}_L \end{pmatrix} \begin{pmatrix} 0 \\ \boldsymbol{g}' \end{pmatrix} \\ &= \boldsymbol{g}'^{\mathrm{H}}\mathbf{R}_L\boldsymbol{g}' ,\end{aligned} \tag{7.71}$$

Similarly,

$$\begin{aligned}\boldsymbol{g}^{\mathrm{H}}\mathbf{R}_{L+1}\boldsymbol{g} &= \begin{pmatrix} \boldsymbol{g}'^{\mathrm{H}} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{R}_L & \boldsymbol{r}_{\mathrm{b},L} \\ \boldsymbol{r}_{\mathrm{b},L}^{\mathrm{T}} & r(0) \end{pmatrix} \begin{pmatrix} \boldsymbol{g}' \\ 0 \end{pmatrix} \\ &= \boldsymbol{g}'^{\mathrm{H}}\mathbf{R}_L\boldsymbol{g}' .\end{aligned} \tag{7.72}$$

Therefore, $\tilde{\boldsymbol{g}}^{\mathrm{H}}\mathbf{R}_{L+1}\tilde{\boldsymbol{g}} = \boldsymbol{g}^{\mathrm{H}}\mathbf{R}_{L+1}\boldsymbol{g}$, and the Schwartz inequality becomes,

$$\left|\tilde{\boldsymbol{g}}^{\mathrm{H}}\mathbf{R}_{L+1}\boldsymbol{g}\right|^2 \le \left(\tilde{\boldsymbol{g}}^{\mathrm{H}}\mathbf{R}_{L+1}\tilde{\boldsymbol{g}}\right)^2 . \tag{7.73}$$

From (7.69) we see that $|\lambda|^2 \le 1$. This completes the proof.

This property allows one easily to ensure that the all-pole system in (7.2) is stable (when the correlation matrix is positive definite) by simply imposing the constraint that the PARCOR coefficients are less than 1 in magnitude. As a result, in speech communication, transmitting PARCOR coefficients is more advantageous than directly transmitting linear predication coefficients.

7.6 Spectral Representation

It is important to understand the link between the spectrum of a speech signal and its prediction coefficients. Let us again take the speech model given in Sect. 7.1,

$$x(k) = \sum_{l=1}^{L} a_l x(k-l) + Gu(k) , \tag{7.74}$$

where we now assume that $u(k)$ is a white random signal with variance $\sigma_u^2 = 1$. Since $x(k)$ is the output of the filter $H(z)$ (see Sect. 7.1), whose input is $u(k)$, its spectrum is [7.11],

$$S_x(\omega) = \left|H(\mathrm{e}^{\mathrm{i}\omega})\right|^2 S_u(\omega) , \tag{7.75}$$

where ω is the angular frequency, $H(\mathrm{e}^{\mathrm{i}\omega})$ is the frequency response of the filter $H(z)$, and $S_u(\omega)$ is the spectrum of u. We have $S_u(\omega) = 1$ (u is white). Using (7.2), we deduce the spectrum of x,

$$\begin{aligned}S_x(\omega) &= \frac{G^2}{|A(\mathrm{e}^{\mathrm{i}\omega})|^2} \\ &= \frac{G^2}{\left|1 - \sum_{l=1}^{L} a_l \mathrm{e}^{-\mathrm{i}l\omega}\right|^2} .\end{aligned} \tag{7.76}$$

Therefore, the spectrum of a speech signal can be modeled by the frequency response of an all-pole filter,

whose elements are the prediction coefficients [7.23–25].

Consider the prediction error signal,

$$e_L(k) = x(k) - \boldsymbol{a}_L^{\mathrm{T}} \boldsymbol{x}(k-1) \,. \tag{7.77}$$

Taking the z-transform of (7.77) and setting $z = \mathrm{e}^{\mathrm{i}\omega}$, we obtain:

$$S_{x,L}(\omega) = \frac{|E_L(\mathrm{e}^{\mathrm{i}\omega})|^2}{|A_L(\mathrm{e}^{\mathrm{i}\omega})|^2} \,, \tag{7.78}$$

where $|E_L(\mathrm{e}^{\mathrm{i}\omega})|^2$ is the spectrum of $e_L(k)$. From Sect. 7.2, we know that, for a large order L, linear prediction tends to whiten the signal, so the power spectrum $|E_L(\mathrm{e}^{\mathrm{i}\omega})|^2$ of the error signal, $e_L(k)$, will tend to be flat. Hence,

$$\lim_{L\to\infty} \left|E_L(\mathrm{e}^{\mathrm{i}\omega})\right|^2 = G^2 \,. \tag{7.79}$$

As a result,

$$\lim_{L\to\infty} S_{x,L}(\omega) = \frac{G^2}{\left|1 - \sum_{l=1}^{\infty} a_l \mathrm{e}^{-\mathrm{i}l\omega}\right|^2} \,. \tag{7.80}$$

This confirms that (7.76) can be a very good approximation of the spectrum of a speech signal, as long as the order of the predictor is large enough.

7.7 Linear Interpolation

Linear interpolation can be seen as a straightforward generalization of forward and backward linear predictions. Indeed, in linear interpolation, we try to predict the value of the sample $x(k-i)$ from its past and future values [7.26, 27]. We define the interpolation error as

$$\begin{aligned} e_i(k) &= x(k-i) - \hat{x}(k-i) \\ &= x(k-i) - \sum_{l=0, l\neq i}^{L} c_{i,l} x(k-l) \\ &= \boldsymbol{c}_i^{\mathrm{T}} \boldsymbol{x}_{L+1}(k), \quad i = 0, 1, \cdots, L, \end{aligned} \tag{7.81}$$

where $\hat{x}(k-i)$ is the interpolated sample,

$$\boldsymbol{c}_i = [-c_{i,0} \;\; -c_{i,1} \;\; \cdots \;\; c_{i,i} \;\; \cdots \;\; -c_{i,L}]^{\mathrm{T}}$$

is a vector of length $L+1$ containing the interpolation coefficients, with $c_{i,i} = 1$, and

$$\boldsymbol{x}_{L+1}(k) = [x(k) \;\; x(k-1) \;\; \cdots \;\; x(k-L)]^{\mathrm{T}} \,.$$

The special cases $i = 0$ and $i = L$ are the forward and backward prediction errors, respectively.

To find the optimal Wiener interpolator, we need to minimize the cost function,

$$\begin{aligned} J_i(\boldsymbol{c}_i) &= E\left\{e_i^2(k)\right\} \\ &= \boldsymbol{c}_i^{\mathrm{T}} \mathbf{R}_{L+1} \boldsymbol{c}_i \,, \end{aligned} \tag{7.82}$$

subject to the constraint

$$\boldsymbol{c}_i^{\mathrm{T}} \boldsymbol{v}_i = c_{i,i} = 1 \,, \tag{7.83}$$

where

$$\boldsymbol{v}_i = [0 \;\; 0 \;\; \cdots \;\; 0 \;\; 1 \;\; 0 \;\; \cdots \;\; 0]^{\mathrm{T}}$$

is a vector of length $L+1$ with its i-th component equal to one and all others equal to zero. By using a Lagrange multiplier, it is easy to see that the solution to this optimization problem is

$$\mathbf{R}_{L+1} \boldsymbol{c}_{\mathrm{o},i} = E_i \boldsymbol{v}_i \,, \tag{7.84}$$

where

$$\begin{aligned} E_i &= \boldsymbol{c}_{\mathrm{o},i}^{\mathrm{T}} \mathbf{R}_{L+1} \boldsymbol{c}_{\mathrm{o},i} \\ &= \frac{1}{\boldsymbol{v}_i^{\mathrm{T}} \mathbf{R}_{L+1}^{-1} \boldsymbol{v}_i} \end{aligned} \tag{7.85}$$

is the interpolation-error power.

From (7.84) we find,

$$\frac{\boldsymbol{c}_{\mathrm{o},i}}{E_i} = \mathbf{R}_{L+1}^{-1} \boldsymbol{v}_i \,, \tag{7.86}$$

hence the i-th column of $\mathbf{R}_{L+1}^{-1}$ is $\boldsymbol{c}_{\mathrm{o},i}/E_i$. We can now see that $\mathbf{R}_{L+1}^{-1}$ can be factorized as follows [7.28]:

$$\begin{aligned} \mathbf{R}_{L+1}^{-1} &= \begin{pmatrix} 1 & -c_{\mathrm{o},1,0} & \cdots & -c_{\mathrm{o},L,0} \\ -c_{\mathrm{o},0,1} & 1 & \cdots & -c_{\mathrm{o},L,1} \\ \vdots & \vdots & \ddots & \vdots \\ -c_{\mathrm{o},0,L} & -c_{\mathrm{o},1,L} & \cdots & 1 \end{pmatrix} \\ &\quad \cdot \begin{pmatrix} 1/E_0 & 0 & \cdots & 0 \\ 0 & 1/E_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1/E_{L-1} \end{pmatrix} \\ &= \mathbf{C}_{\mathrm{o}}^{\mathrm{T}} \mathbf{D}_{\mathrm{e}}^{-1} \,. \end{aligned} \tag{7.87}$$

Furthermore, since $\mathbf{R}_{L+1}^{-1}$ is a symmetric matrix, (7.87) can be written as,

$$\mathbf{R}_{L+1}^{-1} = \begin{pmatrix} 1/E_0 & 0 & \cdots & 0 \\ 0 & 1/E_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1/E_{L-1} \end{pmatrix} \cdot \begin{pmatrix} 1 & -c_{\mathrm{o},0,1} & \cdots & -c_{\mathrm{o},0,L} \\ -c_{\mathrm{o},1,0} & 1 & \cdots & -c_{\mathrm{o},1,L} \\ \vdots & \vdots & \ddots & \vdots \\ -c_{\mathrm{o},L,0} & -c_{\mathrm{o},L,1} & \cdots & 1 \end{pmatrix} = \mathbf{D}_{\mathrm{e}}^{-1}\mathbf{C}_{\mathrm{o}} \,. \tag{7.88}$$

Therefore, we deduce that,

$$\frac{c_{\mathrm{o},i,l}}{E_i} = \frac{c_{\mathrm{o},l,i}}{E_l} \,, \quad i,l = 0,1,\cdots,L \,. \tag{7.89}$$

The first and last columns of $\mathbf{R}_{L+1}^{-1}$ contain, respectively, the normalized forward and backward predictors and all the columns between contain the normalized interpolators.

We are now going to show how the condition number of the correlation matrix depends on the interpolators. The condition number of the matrix $\mathbf{R}_{L+1}$ is defined as [7.29]:

$$\chi(\mathbf{R}_{L+1}) = \|\mathbf{R}_{L+1}\| \left\|\mathbf{R}_{L+1}^{-1}\right\| \,, \tag{7.90}$$

where $\|\cdot\|$ can be any matrix norm. Note that $\chi(\mathbf{R})$ depends on the underlying norm. Let us compute $\chi(\mathbf{R}_{L+1})$ using the Frobenius norm:

$$\begin{aligned} \|\mathbf{R}_{L+1}\|_{\mathrm{F}} &= \left[\mathrm{tr}\left(\mathbf{R}_{L+1}^{\mathrm{T}}\mathbf{R}_{L+1}\right)\right]^{1/2} \\ &= \left[\mathrm{tr}\left(\mathbf{R}_{L+1}^{2}\right)\right]^{1/2} \end{aligned} \tag{7.91}$$

and

$$\left\|\mathbf{R}_{L+1}^{-1}\right\|_{\mathrm{F}} = \left[\mathrm{tr}\left(\mathbf{R}_{L+1}^{-2}\right)\right]^{1/2} \,. \tag{7.92}$$

From (7.86), we have,

$$\frac{\boldsymbol{c}_{\mathrm{o},i}^{\mathrm{T}}\boldsymbol{c}_{\mathrm{o},i}}{E_i^2} = \boldsymbol{v}_i^{\mathrm{T}}\mathbf{R}_{L+1}^{-2}\boldsymbol{v}_i \,, \tag{7.93}$$

which implies that,

$$\begin{aligned} \sum_{i=0}^{L} \frac{\boldsymbol{c}_{\mathrm{o},i}^{\mathrm{T}}\boldsymbol{c}_{\mathrm{o},i}}{E_i^2} &= \sum_{i=0}^{L} \boldsymbol{v}_i^{\mathrm{T}}\mathbf{R}_{L+1}^{-2}\boldsymbol{v}_i \\ &= \mathrm{tr}\left(\mathbf{R}_{L+1}^{-2}\right) \,. \end{aligned} \tag{7.94}$$

Also, we can easily check that,

$$\mathrm{tr}\left(\mathbf{R}_{L+1}^{2}\right) = (L+1)r^2(0) + 2\sum_{l=1}^{L}(L+1-l)r^2(l) \,. \tag{7.95}$$

Therefore, the square of the condition number of the correlation matrix associated with the Frobenius norm is

$$\chi_{\mathrm{F}}^2(\mathbf{R}_{L+1}) = \left[(L+1)r^2(0) + 2\sum_{l=1}^{L}(L+1-l)r^2(l)\right] \times \sum_{i=0}^{L} \frac{\boldsymbol{c}_{\mathrm{o},i}^{\mathrm{T}}\boldsymbol{c}_{\mathrm{o},i}}{E_i^2} \,. \tag{7.96}$$

Some other interesting relations between the forward predictors and the condition number can be found in [7.30].

To conclude this section, we would like to let readers know that several algorithms exist to compute the optimal predictors efficiently, see for example, [7.31] and [7.32]. All these algorithms are based on Levinson–Durbin recursions.

7.8 Line Spectrum Pair Representation

Line spectrum pair (LSP) representation, first introduced by *Itakura* [7.33], is a more-robust way to represent the coefficients of linear predictive models. The LSP polynomials have some very interesting properties shown in [7.34].

A polynomial $P(z)$ of order L is said to be symmetric if

$$P(z) = z^{-L}P(z^{-1}) \tag{7.97}$$

and a polynomial $Q(z)$ is antisymmetric if

$$Q(z) = -z^{-L}Q(z^{-1}) \,. \tag{7.98}$$

Let

$$A(z) = 1 - a_1 z^{-1} - a_2 z^{-2} - \cdots - a_L z^{-L} \tag{7.99}$$

be the optimal polynomial predictor of order L. It is well known that in speech compression the coefficients of this polynomial are inappropriate for quantization because of their relatively large dynamic range and also because, as stated earlier, quantization can change a stable LPC filter into an unstable one [7.35]. From (7.99), we can construct two artificial $(L+1)$-th-order (symmetric and antisymmetric) polynomials by setting the $(L+1)$-th

reflection coefficient, κ_{L+1}, to be $+1$ and -1. These two cases correspond, respectively, to an entirely closed or to an entirely open end at the last section of an acoustic tube of $L+1$ piecewise-uniform sections [7.35],

$$P(z) = A(z) + z^{-L}A(z^{-1}) , \tag{7.100}$$

$$Q(z) = A(z) - z^{-L}A(z^{-1}) . \tag{7.101}$$

The polynomial $A(z)$ can be easily reconstructed from $P(z)$ and $Q(z)$ by

$$A(z) = \frac{1}{2}[P(z) + Q(z)] . \tag{7.102}$$

It was proved in [7.36] and [7.34] that the LSP polynomials, $P(z)$ and $Q(z)$, have the following important properties:

- all zeros of LSP polynomials are on the unit circle,
- the zeros of $P(z)$ and $Q(z)$ are interlaced, and
- the minimum-phase property of $A(z)$ can be easily preserved if the first two properties are intact after quantization.

Now, define the two prediction error signals:

$$e^{+}(k) = x(n) - \frac{1}{2}[\boldsymbol{x}(n-1) + \boldsymbol{x}(n)]^{\mathrm{T}}\boldsymbol{a}^{+}, \tag{7.103}$$

$$e^{-}(k) = x(n) - \frac{1}{2}[\boldsymbol{x}(n-1) - \boldsymbol{x}(n)]^{\mathrm{T}}\boldsymbol{a}^{-} . \tag{7.104}$$

It is shown in [7.37] and [7.38] that the LSP polynomials, whose trivial zeroes have been removed, are equivalent to the two optimal Wiener predictors $\boldsymbol{a}_{\mathrm{o}}^{+}$ and $\boldsymbol{a}_{\mathrm{o}}^{-}$. This is easy to see if we rewrite, $e^{+}(k)$ for example as

$$\begin{aligned} e^{+}(k) &= \left[1 \quad -\frac{\boldsymbol{a}^{+T}}{2}\right]\boldsymbol{x}_{L+1}(n) \\ &\quad + \left[-\frac{\boldsymbol{a}^{+T}}{2}\ 0\right]\boldsymbol{x}_{L+1}(n) \\ &= \boldsymbol{g}^{\mathrm{T}}\mathbf{I}_{\mathrm{d}}^{\mathrm{T}}\boldsymbol{x}_{L+1}(n) , \end{aligned} \tag{7.105}$$

where

$$\mathbf{I}_{\mathrm{d}} = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 1 \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \tag{7.106}$$

and

$$\boldsymbol{g} = \begin{pmatrix} 1 \\ -\dfrac{\boldsymbol{a}^{+}}{2} \end{pmatrix} . \tag{7.107}$$

By minimizing the MSE, $E\{e^{+2}(k)\}$, with respect to $\boldsymbol{g}$, with the constraint $\boldsymbol{g}^{\mathrm{T}}\boldsymbol{v}_1 = 1$, one can find the most important results. For readers who are interested in more details on the properties of LSP polynomials, we recommend the paper by *Bäckström* and *Magi* [7.39].

7.9 Multichannel Linear Prediction

Multichannel linear prediction can be very useful in stereo or multichannel speech compression. In an increasing number of speech or audio applications, we have at least two channels available, which are often highly correlated with each other. Therefore, it makes sense to take this interchannel correlation into account in order to obtain more-efficient compression schemes. Multichannel linear prediction is the best way to do this.

Let

$$\chi(k) = \left[x_1(k) \ \ x_2(k) \ \cdots \ x_M(k)\right]^{\mathrm{T}}$$

be a real, zero-mean, stationary M-channel time series. We define the multichannel forward prediction error vector as,

$$\begin{aligned} \boldsymbol{e}_{\mathrm{f},L}(k) &= \chi(k) - \hat{\chi}(k) \\ &= \chi(k) - \sum_{l=1}^{L}\mathbf{A}_{L,l}\chi(k-l) \\ &= \chi(k) - \mathbf{A}_{L}^{\mathrm{T}}\boldsymbol{x}(k-1) , \end{aligned} \tag{7.108}$$

where

$$\mathbf{A}_L = [\mathbf{A}_{L,1} \ \ \mathbf{A}_{L,2} \ \cdots \ \mathbf{A}_{L,L}]^{\mathrm{T}}$$

is the forward predictor matrix of size $ML \times M$, each one of the square matrices $\mathbf{A}_{L,l}$ is of size $M \times M$, and

$$\boldsymbol{x}(k-1) = [\chi^{\mathrm{T}}(k-1) \ \ \chi^{\mathrm{T}}(k-2) \ \cdots \ \chi^{\mathrm{T}}(k-L)]^{\mathrm{T}}$$

is a vector of length ML. (For convenience, some of the notation used in this section is the same as that in the previous sections.)

To derive the optimal Wiener forward predictors, we need to minimize the MSE,

$$J_{\mathrm{f}}(\mathbf{A}_L) = E\left\{\boldsymbol{e}_{\mathrm{f},L}^{\mathrm{T}}(k)\boldsymbol{e}_{\mathrm{f},L}(k)\right\} . \tag{7.109}$$

We find the multichannel Wiener–Hopf equations:

$$\mathbf{R}_L\mathbf{A}_{\mathrm{o},L} = \mathbf{R}_{\mathrm{f}}(1/L) , \tag{7.110}$$

where

$$\begin{aligned}\mathbf{R}_L &= E\{\boldsymbol{x}(k-1)\boldsymbol{x}^{\mathrm{T}}(k-1)\} \qquad (7.111)\\ &= E\{\boldsymbol{x}(k)\boldsymbol{x}^{\mathrm{T}}(k)\}\\ &= \begin{pmatrix} \mathbf{R}(0) & \mathbf{R}(1) & \cdots & \mathbf{R}(L-1)\\ \mathbf{R}^{\mathrm{T}}(1) & \mathbf{R}(0) & \cdots & \mathbf{R}(L-2)\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{R}^{\mathrm{T}}(L-1) & \mathbf{R}^{\mathrm{T}}(L-2) & \cdots & \mathbf{R}(0)\end{pmatrix} \qquad (7.112)\end{aligned}$$

is the block-Toeplitz covariance matrix of size ML×ML,

$$\begin{aligned}&\mathbf{R}(l) = E\{\chi(k)\chi^{\mathrm{T}}(k-l)\}, \quad l = 0, 1, \cdots, L-1 ,\\ &\mathbf{R}(-l) = E\{\chi(k-l)\chi^{\mathrm{T}}(k)\} = \mathbf{R}^{\mathrm{T}}(l) ,\end{aligned}$$

and

$$\begin{aligned}\mathbf{R}_{\mathrm{f}}(1/L) &= [\mathbf{R}(1)\ \ \mathbf{R}(2)\ \cdots\ \mathbf{R}(L)]^{\mathrm{T}}\\ &= E\{\boldsymbol{x}(k-1)\chi^{\mathrm{T}}(k)\}\end{aligned}$$

is the intercorrelation matrix of size ML×M.

Using the augmented block-Toeplitz covariance matrix of size (ML+M)×(ML+M):

$$\mathbf{R}_{L+1} = \begin{pmatrix} \mathbf{R}(0) & \mathbf{R}_{\mathrm{f}}^{\mathrm{T}}(1/L)\\ \mathbf{R}_{\mathrm{f}}(1/L) & \mathbf{R}_L\end{pmatrix} , \tag{7.113}$$

we deduce the augmented multichannel Wiener–Hopf equations:

$$\mathbf{R}_{L+1}\begin{pmatrix}\mathbf{I}_{M\times M}\\ -\mathbf{A}_{\mathrm{o},L}\end{pmatrix} = \begin{pmatrix}\mathbf{E}_{\mathrm{f},L}\\ \mathbf{0}_{\mathrm{ML}\times M}\end{pmatrix} , \tag{7.114}$$

where $\mathbf{I}_{M\times M}$ is the identity matrix of size $M\times M$ and

$$\begin{aligned}\mathbf{E}_{\mathrm{f},L} &= E\left\{\boldsymbol{e}_{\mathrm{f,o},L}(k)\boldsymbol{e}_{\mathrm{f,o},L}^{\mathrm{T}}(k)\right\}\\ &= \mathbf{R}(0) - \mathbf{R}_{\mathrm{f}}^{\mathrm{T}}(1/L)\mathbf{A}_{\mathrm{o},L} \qquad (7.115)\end{aligned}$$

is the forward error covariance matrix of size $M\times M$, with

$$\boldsymbol{e}_{\mathrm{f,o},L}(k) = \chi(k) - \mathbf{A}_{\mathrm{o},L}^{\mathrm{T}}\boldsymbol{x}(k-1) . \tag{7.116}$$

We will proceed with the same philosophy to derive important equations for the multichannel backward prediction. We define the multichannel backward prediction error vector as

$$\begin{aligned}\boldsymbol{e}_{\mathrm{b},L}(k) &= \chi(k-L) - \hat{\chi}(k-L)\\ &= \chi(k-L) - \sum_{l=1}^{L}\mathbf{B}_{L,l}\chi(k-l+1)\\ &= \chi(k-L) - \mathbf{B}_L^{\mathrm{T}}\boldsymbol{x}(k) , \qquad (7.117)\end{aligned}$$

where

$$\mathbf{B}_L = [\mathbf{B}_{L,1}\ \ \mathbf{B}_{L,2}\ \cdots\ \mathbf{B}_{L,L}]^{\mathrm{T}}$$

is the backward predictor matrix of size ML×M with each one of the square submatrices $\mathbf{B}_{L,l}$ being of size $M\times M$.

The minimization of the MSE,

$$J_{\mathrm{b}}(\mathbf{B}_L) = E\left\{\boldsymbol{e}_{\mathrm{b},L}^{\mathrm{T}}(k)\boldsymbol{e}_{\mathrm{b},L}(k)\right\} , \tag{7.118}$$

leads to the multichannel Wiener–Hopf equations for the backward prediction:

$$\mathbf{R}_L\mathbf{B}_{\mathrm{o},L} = \mathbf{R}_{\mathrm{b}}(1/L) , \tag{7.119}$$

where

$$\begin{aligned}\mathbf{R}_{\mathrm{b}}(1/L) &= E\{\boldsymbol{x}(k)\chi^{\mathrm{T}}(k-L)\} \qquad (7.120)\\ &= [\mathbf{R}^{\mathrm{T}}(L)\ \ \mathbf{R}^{\mathrm{T}}(L-1)\ \cdots\ \mathbf{R}^{\mathrm{T}}(1)]^{\mathrm{T}} .\end{aligned}$$

By using the augmented block-Toeplitz covariance matrix:

$$\mathbf{R}_{L+1} = \begin{pmatrix}\mathbf{R}_L & \mathbf{R}_{\mathrm{b}}(1/L)\\ \mathbf{R}_{\mathrm{b}}^{\mathrm{T}}(1/L) & \mathbf{R}(0)\end{pmatrix} , \tag{7.121}$$

we find the augmented multichannel Wiener–Hopf equations:

$$\mathbf{R}_{L+1}\begin{pmatrix}-\mathbf{B}_{\mathrm{o},L}\\ \mathbf{I}_{M\times M}\end{pmatrix} = \begin{pmatrix}\mathbf{0}_{\mathrm{ML}\times M}\\ \mathbf{E}_{\mathrm{b},L}\end{pmatrix} , \tag{7.122}$$

where

$$\begin{aligned}\mathbf{E}_{\mathrm{b},L} &= E\left\{\boldsymbol{e}_{\mathrm{b,o},L}(k)\boldsymbol{e}_{\mathrm{b,o},L}^{\mathrm{T}}(k)\right\}\\ &= \mathbf{R}(0) - \mathbf{R}_{\mathrm{b}}^{\mathrm{T}}(1/L)\mathbf{B}_{\mathrm{o},L} \qquad (7.123)\end{aligned}$$

is the backward error covariance matrix of size $M\times M$, with

$$\boldsymbol{e}_{\mathrm{b,o},L}(k) = \chi(k-L) - \mathbf{B}_{\mathrm{o},L}^{\mathrm{T}}\boldsymbol{x}(k) . \tag{7.124}$$

To solve the multichannel Wiener–Hopf equations efficiently, we need to derive some important relations [7.40]. Consider the following system,

$$\begin{pmatrix} \mathbf{R}_L & \mathbf{R}_{\mathrm{b}}(1/L) \\ \mathbf{R}_{\mathrm{b}}^{\mathrm{T}}(1/L) & \mathbf{R}(0) \end{pmatrix} \begin{pmatrix} \mathbf{I}_{M\times M} \\ -\mathbf{A}_{\mathrm{o},L-1} \\ \mathbf{0}_{M\times M} \end{pmatrix} = \begin{pmatrix} \mathbf{E}_{\mathrm{f},L-1} \\ \mathbf{0}_{(ML-M)\times M} \\ \mathbf{K}_{\mathrm{f},L} \end{pmatrix} , \tag{7.125}$$

where

$$\mathbf{K}_{\mathrm{f},L} = \mathbf{R}^{\mathrm{T}}(L) - \mathbf{R}_{\mathrm{b}}^{\mathrm{T}}(1/L-1)\mathbf{A}_{\mathrm{o},L-1} . \tag{7.126}$$

Consider the other system,

$$\begin{pmatrix} \mathbf{R}(0) & \mathbf{R}_{\mathrm{f}}^{\mathrm{T}}(1/L) \\ \mathbf{R}_{\mathrm{f}}(1/L) & \mathbf{R}_L \end{pmatrix} \begin{pmatrix} \mathbf{0}_{M\times M} \\ -\mathbf{B}_{\mathrm{o},L-1} \\ \mathbf{I}_{M\times M} \end{pmatrix} = \begin{pmatrix} \mathbf{K}_{\mathrm{b},L} \\ \mathbf{0}_{(ML-M)\times M} \\ \mathbf{E}_{\mathrm{b},L-1} \end{pmatrix} , \tag{7.127}$$

where

$$\mathbf{K}_{\mathrm{b},L} = \mathbf{R}(L) - \mathbf{R}_{\mathrm{f}}^{\mathrm{T}}(1/L-1)\mathbf{B}_{\mathrm{o},L-1} . \tag{7.128}$$

If we post-multiply both sides of (7.127) by $\mathbf{E}_{\mathrm{b},L-1}^{-1}\mathbf{K}_{\mathrm{f},L}$, we get:

$$\mathbf{R}_{L+1} \begin{pmatrix} \mathbf{0}_{M\times M} \\ -\mathbf{B}_{\mathrm{o},L-1} \\ \mathbf{I}_{M\times M} \end{pmatrix} \mathbf{E}_{\mathrm{b},L-1}^{-1}\mathbf{K}_{\mathrm{f},L} = \begin{pmatrix} \mathbf{K}_{\mathrm{b},L}\mathbf{E}_{\mathrm{b},L-1}^{-1}\mathbf{K}_{\mathrm{f},L} \\ \mathbf{0}_{(ML-M)\times M} \\ \mathbf{K}_{\mathrm{f},L} \end{pmatrix} . \tag{7.129}$$

Subtracting (7.129) from (7.125) and identifying the resulting system with the augmented multichannel Wiener–Hopf equations for forward prediction [eq. (7.114)], we deduce the two recursions:

$$\mathbf{E}_{\mathrm{f},L} = \mathbf{E}_{\mathrm{f},L-1} - \mathbf{K}_{\mathrm{b},L}\mathbf{E}_{\mathrm{b},L-1}^{-1}\mathbf{K}_{\mathrm{f},L} , \tag{7.130}$$

$$\mathbf{A}_{\mathrm{o},L} = \begin{pmatrix} \mathbf{A}_{\mathrm{o},L-1} \\ \mathbf{0}_{M\times M} \end{pmatrix} - \begin{pmatrix} \mathbf{B}_{\mathrm{o},L-1} \\ -\mathbf{I}_{M\times M} \end{pmatrix} \mathbf{E}_{\mathrm{b},L-1}^{-1}\mathbf{K}_{\mathrm{f},L} . \tag{7.131}$$

Similarly, if we post-multiply both sides of (7.125) by $\mathbf{E}_{\mathrm{f},L-1}^{-1}\mathbf{K}_{\mathrm{b},L}$, we obtain:

$$\mathbf{R}_{L+1} \begin{pmatrix} \mathbf{I}_{M\times M} \\ -\mathbf{A}_{\mathrm{o},L-1} \\ \mathbf{0}_{M\times M} \end{pmatrix} \mathbf{E}_{\mathrm{f},L-1}^{-1}\mathbf{K}_{\mathrm{b},L} = \begin{pmatrix} \mathbf{K}_{\mathrm{b},L} \\ \mathbf{0}_{(ML-M)\times M} \\ \mathbf{K}_{\mathrm{f},L}\mathbf{E}_{\mathrm{f},L-1}^{-1}\mathbf{K}_{\mathrm{b},L} \end{pmatrix} . \tag{7.132}$$

Subtracting (7.132) from (7.127) and identifying the resulting system with the augmented multichannel Wiener–Hopf equations for backward prediction (7.122), we deduce the two recursions:

$$\mathbf{E}_{\mathrm{b},L} = \mathbf{E}_{\mathrm{b},L-1} - \mathbf{K}_{\mathrm{f},L}\mathbf{E}_{\mathrm{f},L-1}^{-1}\mathbf{K}_{\mathrm{b},L} , \tag{7.133}$$

$$\mathbf{B}_{\mathrm{o},L} = \begin{pmatrix} \mathbf{0}_{M\times M} \\ \mathbf{B}_{\mathrm{o},L-1} \end{pmatrix} - \begin{pmatrix} -\mathbf{I}_{M\times M} \\ \mathbf{A}_{\mathrm{o},L-1} \end{pmatrix} \mathbf{E}_{\mathrm{f},L-1}^{-1}\mathbf{K}_{\mathrm{b},L} . \tag{7.134}$$

Relations (7.130), (7.131), (7.133), and (7.134) were independently discovered by *Whittle* [7.41] and *Wiggins* and *Robinson* [7.42].

Another important relation needs to be found. Indeed, using (7.116) and (7.124), we can easily verify,

$$E\left\{\boldsymbol{e}_{\mathrm{f,o},L-1}(k)\boldsymbol{e}_{\mathrm{b,o},L-1}^{\mathrm{T}}(k-1)\right\} = \mathbf{K}_{\mathrm{b},L} , \tag{7.135}$$

$$E\left\{\boldsymbol{e}_{\mathrm{b,o},L-1}(k-1)\boldsymbol{e}_{\mathrm{f,o},L-1}^{\mathrm{T}}(k)\right\} = \mathbf{K}_{\mathrm{f},L} , \tag{7.136}$$

which implies that,

$$\mathbf{K}_{\mathrm{b},L} = \mathbf{K}_{\mathrm{f},L}^{\mathrm{T}} . \tag{7.137}$$

Table 7.2 summarizes the Levinson–Wiggins–Robinson algorithm [7.42–44], which is a generalization of the Levinson–Durbin algorithm to the multichannel case.

Table 7.2 Levinson–Wiggins–Robinson algorithm

Initialization: $\mathbf{E}_{\mathrm{f},0} = \mathbf{E}_{\mathrm{b},0} = \mathbf{R}(0)$

For $1 \le l \le L$

$$\mathbf{K}_{\mathrm{b},l} = \mathbf{R}(l) - \mathbf{R}_{\mathrm{f}}^{\mathrm{T}}(1:l-1)\mathbf{B}_{\mathrm{o},l-1}$$

$$\mathbf{A}_{\mathrm{o},l} = \begin{bmatrix} \mathbf{A}_{\mathrm{o},l-1} \\ \mathbf{0}_{M\times M} \end{bmatrix} - \begin{bmatrix} \mathbf{B}_{\mathrm{o},l-1} \\ -\mathbf{I}_{M\times M} \end{bmatrix} \mathbf{E}_{\mathrm{b},l-1}^{-1}\mathbf{K}_{\mathrm{b},l}^{\mathrm{T}}$$

$$\mathbf{B}_{\mathrm{o},l} = \begin{bmatrix} \mathbf{0}_{M\times M} \\ \mathbf{B}_{\mathrm{o},l-1} \end{bmatrix} - \begin{bmatrix} -\mathbf{I}_{M\times M} \\ \mathbf{A}_{\mathrm{o},l-1} \end{bmatrix} \mathbf{E}_{\mathrm{f},l-1}^{-1}\mathbf{K}_{\mathrm{b},l}$$

$$\mathbf{E}_{\mathrm{f},l} = \mathbf{E}_{\mathrm{f},l-1} - \mathbf{K}_{\mathrm{b},l}\mathbf{E}_{\mathrm{b},l-1}^{-1}\mathbf{K}_{\mathrm{b},l}^{\mathrm{T}}$$

$$\mathbf{E}_{\mathrm{b},l} = \mathbf{E}_{\mathrm{b},l-1} - \mathbf{K}_{\mathrm{b},l}^{\mathrm{T}}\mathbf{E}_{\mathrm{f},l-1}^{-1}\mathbf{K}_{\mathrm{b},l}$$

7.10 Conclusions

In this chapter, we have tried to present the most important results in linear prediction for speech. We have explained the principle of forward linear prediction and have shown that the optimal prediction error signal tends to be a white signal. We have extended the principle of forward linear prediction to backward linear prediction and derived the Cholesky factorization of the inverse correlation matrix. We have developed the classical Levinson–Durbin algorithm, which is a very efficient way to solve the Wiener–Hopf equations for the forward and backward prediction coefficients. We have explained the idea behind the lattice predictor. We have shown how the spectrum of a speech signal can easily be estimated thanks to the prediction coefficients. We have given some notions of linear interpolation and have demonstrated how the condition number of the correlation matrix is related to the optimal interpolators. We have also presented some notions of line spectrum pair polynomials. Finally, in the last section, we have generalized some of these ideas to the multichannel case.

References

7.1 B.S. Atal: The history of linear prediction, IEEE Signal Proc. Mag. **23**, 154–161 (2006)

7.2 L.R. Rabiner, R.W. Schaffer: *Digital Processing of Speech Signals* (Prentice-Hall, Englewood Cliffs 1976)

7.3 J.D. Markel, A.H. Gray Jr.: *Linear Prediction of Speech* (Springer, New York 1976)

7.4 J. Makhoul: Linear prediction: a tutorial review, Proc. IEEE **63**, 561–580 (1975)

7.5 J.W. Picone: Signal modeling techniques in speech recognition, Proc. IEEE **81**, 1215–1247 (1993)

7.6 S.R. Quackenbush, T.P. Barnwell, M.A. Clements: *Objective Measures of Speech Quality* (Prentice-Hall, Englewood Cliffs 1988)

7.7 Y. Huang, J. Benesty, J. Chen: *Acoustic MIMO Signal Processing* (Springer, New York 2006)

7.8 F. Itakura, S. Saito: A statistical method for estimation of speech spectral density and formant frequencies, Electron. Commun. Jpn. **53**(A), 36–43 (1970)

7.9 G. Chen, S.N. Koh, I.Y. Soon: Enhanced Itakura measure incorporating masking properties of human auditory system, Signal Process. **83**, 1445–1456 (2003)

7.10 M.G. Bellanger: *Adaptive Digital Filters and Signal analysis* (Marcel Dekker, New York 1987)

7.11 S. Haykin: *Adaptive Filter Theory*, 4th edn. (Prentice-Hall, Upper Saddle River 2002)

7.12 C.W. Therrien: On the relation between triangular matrix decomposition and linear prediction, Proc. IEEE **71**, 1459–1460 (1983)

7.13 N. Levinson: The Wiener RMS (root mean square) error criterion in filter design and prediction, J. Math. Phys. **25**(4), 261–278 (1947), Also Appendix B, in N. Wiener, Extrapolation, Interpolation and Smoothing of Stationary Time Series (MIT, Cambridge 1949)

7.14 J. Durbin: Efficient estimation of parameters in moving-average models, Biometrika **46**, 306–316 (1959), Parts 1 and 2

7.15 J. Durbin: The fitting of time-series models, Rev. Inst. Int. Stat. **28**(3), 233–243 (1960)

7.16 J.P. Burg: *Maximum Entropy Spectral Analysis Ph.D. Dissertation* (Stanford Univ., Stanford 1975)

7.17 P. Delsarte, Y.V. Genin: The split Levinson algorithm, IEEE Trans. Acoust. Speech **ASSP-34**, 470–478 (1986)

7.18 R. Kumar: A fast algorithm for solving a Toeplitz system of equations, IEEE Trans. Acoust. Speech **ASSP-33**, 254–265 (1985)

7.19 J. Makhoul: Stable and efficient lattice methods for linear prediction, IEEE Trans. Acoust. Speech **ASSP-25**, 423–428 (1977)

7.20 F. Itakura, S. Saito: Digital filtering techniques for speech analysis and synthesis, Proc. 7th Int. Conf. Acoust. **25**(C-1), 261–264 (1971)

7.21 S.W. Lang, J.H. McClellan: A simple proof of stability for all-pole linear prediction models, Proc. IEEE **67**, 860–861 (1979)

7.22 M.H. Hayes: *Statistical Digital Signal Processing and Modeling* (Wiley, New York 1996)

7.23 J. Makhoul: Spectral analysis of speech by linear prediction, IEEE Trans. Acoust. Speech **AU-21**(3), 140–148 (1973)

7.24 S.M. Kay, S.L. Marple Jr.: Spectrum analysis – a modern perspective, Proc. IEEE **69**, 1380–1419 (1981)

7.25 J. M.Cadzow: Spectral estimation: an overdetermined rational model equation approach, Proc. IEEE **70**, 907–939 (1982)

7.26 S. Kay: Some results in linear interpolation theory, IEEE Trans. Acoust. Speech **ASSP-31**, 746–749 (1983)

7.27 B. Picinbono, J.-M. Kerilis: Some properties of prediction and interpolation errors, IEEE Trans. Acoust. Speech **ASSP-36**, 525–531 (1988)

7.28 J. Benesty, T. Gaensler: New insights into the RLS algorithm, EURASIP J. Appl. Si. Pr. **2004**, 331–339 (2004)

7.29 G.H. Golub, C.F. Van Loan: *Matrix Computations* (Johns Hopkins Univ. Press, Baltimore 1996)

7.30 J. Benesty, T. Gaensler: Computation of the condition number of a non-singular symmetric Toeplitz matrix with the Levinson-Durbin algorithm, IEEE Trans. Signal Proces. **54**, 2362–2364 (2006)

7.31 C.K. Coursey, J.A. Stuller: Linear interpolation lattice, IEEE Trans. Signal Proces. **39**, 965–967 (1991)

7.32 M.R.K. Khansari, A. Leon-Garcia: A fast algorithm for optimal linear interpolation, IEEE Trans. Signal Process. **41**, 2934–2937 (1993)

7.33 F. Itakura: Line spectrum representation of linear predictive coefficients of speech signal, J. Acoust. Soc. Am. **57**(1), 35 (1975)

7.34 F.K. Soong, B.-H. Juang: Line spectrum pair (LSP) and speech data compression, Proc. ICASSP (1984) pp. 1.10.1–1.10.4

7.35 F.K. Soong, B.-H. Juang: Optimal quantization of LSP paramaters, IEEE Trans. Speech Audio Process. **1**, 15–24 (1993)

7.36 S. Sagayama: Stability condition of LSP speech synthesis didgital filter, Proc. Acoust. Soc. Jpn. (1982) pp. 153–154, (in Japanese)

7.37 B. Kleijn, T. Bäckström, P. Alku: On line spectral frequencies, IEEE Signal Proc. Lett. **10**, 75–77 (2003)

7.38 T. Bäckström, P. Alku, T. Paatero, B. Kleijn: A time-domain interpretation for the LSP decomposition, IEEE Trans. Speech Audio Process. **12**, 554–560 (2004)

7.39 T. Bäckström, C. Magi: Properties of line spectrum pair polynomials – A review, Elsevier Signal Process. **86**, 3286–3298 (2006)

7.40 T. Kailath: A view of three decades of linear filtering theory, IEEE Trans. Inf. Theory **IT-20**, 146–181 (1974)

7.41 P. Whittle: On the fitting of multivariate autoregressions and the approximate canonical factorization of a spectral density matrix, Biometrika **50**, 129–134 (1963)

7.42 R.A. Wiggins, E.A. Robinson: Recursive solution to the multichannel filtering problem, J. Geophys. Res. **70**, 1885–1891 (1965)

7.43 O.N. Strand: Multichannel complex maximum entropy (autoregressive) spectral analysis, IEEE T. Automat. Contr. **AC-22**, 634–640 (1977)

7.44 P. Delsarte, Y.V. Genin: Multichannel singular predictor polynomials, IEEE Trans. Circuits Syst. **35**, 190–200 (1988)

10. Pitch and Voicing Determination of Speech with an Extension Toward Music Signals

W. J. Hess

This chapter reviews selected methods for pitch determination of speech and music signals. As both these signals are time variant we first define what is subsumed under the term *pitch*. Then we subdivide pitch determination algorithms (PDAs) into short-term analysis algorithms, which apply some spectral transform and derive pitch from a frequency or lag domain representation, and time-domain algorithms, which analyze the signal directly and apply structural analysis or determine individual periods from the first partial or compute the instant of glottal closure in speech. In the 1970s, when many of these algorithms were developed, the main application in speech technology was the vocoder, whereas nowadays prosody recognition in speech understanding systems and high-accuracy pitch period determination for speech synthesis corpora are emphasized. In musical acoustics, pitch determination is applied in melody recognition or automatic musical transcription, where we also have the problem that several pitches can exist simultaneously.

Pitch and voicing determination of speech signals are the two subproblems of voice source analysis. In voiced speech, the vocal cords vibrate in a quasiperiodic way. Speech segments with voiceless excitation are generated by turbulent air flow at a constriction or by the release of a closure in the vocal tract. The parameters we have to determine are the manner of excitation, i. e., the presence of a voiced excitation and the presence of a voiceless excitation, a problem we will refer to as *voicing determination* and, for the segments of the speech signal in which a voiced excitation is present, the rate of vocal cord vibration, which is usually referred to as *pitch determination* or *fundamental frequency determination* in the literature.

Unlike the analysis of vocal-tract parameters, where a number of independent and equivalent representations are possible, there is no alternative to the parameters pitch and voicing, and the quality of a synthesized signal critically depends on their reliable and accurate determination. This chapter presents a selection of the methods applied in pitch and voicing determination. The emphasis, however, is on pitch determination.

Over the last two decades, the task of fundamental frequency determination has become increasingly popular in musical acoustics as well. In the beginning the methodology was largely imported from the speech community, but then the musical acoustics community developed algorithms and applications of their own, which in turn became increasingly interesting to the speech communication area. Hence it appears justified to include the aspect of fundamental frequency determination of music signals and to present some of the methods and specific problems of this area. One specific problem is multipitch determination from polyphonic signals, a problem that might also occur in speech when we have to separate two or more simultaneously speaking voices.

Pitch determination has a rather long history which goes back even beyond the times of vocoding. Literally hundreds of pitch determination algorithms (PDAs) have been developed. The most important developments leading to today's state of the art were made in the 1960s and 1970s; most of the methods that are briefly reviewed in this chapter were extensively discussed during this period [10.1]. Since then, least-squares and other statistical methods, particularly in connection with sinusoidal models [10.2], entered the domain. A number of known methods were improved and refined, whereas other solutions that required an amount of computational effort that appeared prohibitive at the time the algorithm was first developed were revived. With the widespread use of databases containing many labeled and processed speech data, it has nowadays also become possible to thoroughly evaluate the performance of the algorithms.

The bibliography in [10.1], dating from 1983, includes about 2000 entries. To give a complete overview of the more-recent developments, at least another 1000 bibliographic entries would have to be added. It goes without saying that this is not possible here given the space limitations. So we will necessarily have to present a selection, and many important contributions cannot be described.

The remainder of this chapter is organized as follows. In Sect. 10.1 the problems of pitch and voicing determination are described, definitions of what is subsumed under the term *pitch* are given, and the various PDAs are grossly categorized. Sections 10.2 and 10.3 give a more-detailed description of selected PDAs. Section 10.4 shortly reviews a selection of voicing determination algorithms (VDAs); Sect. 10.5 deals with questions of error analysis and evaluation. Selected applications are discussed in Sect. 10.6, and Sect. 10.7 finally presents a couple of new developments, such as determining the instant of glottal closure or processing signals that contain more than one pitch, such as polyphonic music.

10.1 Pitch in Time-Variant Quasiperiodic Acoustic Signals

10.1.1 Basic Definitions

Pitch, i. e., the fundamental frequency F_0 and fundamental period T_0 of a (quasi)periodic signal, can be measured in many ways. If a signal is completely stationary and periodic, all these strategies – provided they operate correctly – lead to identical results. Since both speech and musical signals, however, are nonstationary and time variant, aspects of each strategy such as the starting point of the measurement, the length of the measuring interval, the way of averaging (if any), or the operating domain (time, frequency, lag etc.) start to influence the results and may lead to estimates that differ from algorithm to algorithm even if all these results are *correct* and *accurate*. Before entering a discussion on individual methods and applications, we must therefore have a look at the parameter pitch and provide a clear definition of what should be measured and what is actually measured.

A word on terminology first. There are three points of view for looking at such a problem of acoustic signal processing [10.3]: the *production*, the *signal-processing*, and the *perception* points of view. For pitch determination of speech, the production point of view is obviously oriented toward phonation in the human larynx; we will thus have to start from a time-domain representation of the waveform as a train of laryngeal pulses. If a pitch determination algorithm (PDA) works in a speech-production oriented way, it measures individual *laryngeal excitation cycles* or, if some averaging is performed, the *rate of vocal-fold vibration*. The signal-processing point of view, which can be applied

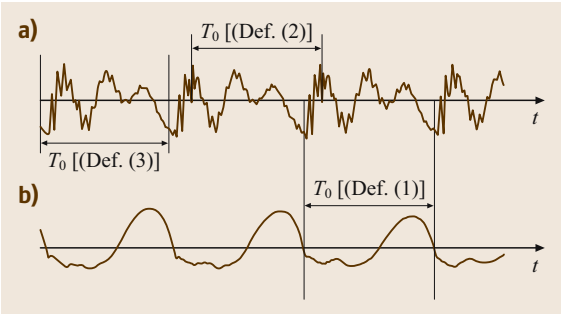


Fig. 10.1a,b Time-domain definitions of T_0. **(a)** Speech signal (a couple of periods), **(b)** glottal waveform (reconstructed). For further detail, see the text

to any acoustic signal, means that (quasi)periodicity or at least cyclic behavior is observed, and that the task is to extract those features that best represent this periodicity. The pertinent terms are *fundamental frequency* and *fundamental period*. If individual cycles are determined, we may (somewhat inconsistently) speak of *pitch periods* or simply of *periods*. The perception point of view leads to a frequency-domain representation since pitch sensation primarily corresponds to a frequency [10.4, 5] even if a time-domain mechanism is involved [10.6]. This point of view is associated with the original meaning of the term *pitch*. Yet the term *pitch* has consistently been used as some kind of common denominator and a general name for all those terms, at least in the technical literature on speech [10.7]. In the following, we will therefore use the term *pitch* in this wider sense, even for musical signals.

When we proceed from production to perception, we arrive at five basic definitions of pitch that apply to speech signals and could read as follows ([10.1, 8, 9]; Fig. 10.1):

1. *T_0 is defined as the elapsed time between two successive laryngeal pulses. Measurement starts at a well-specified point within the glottal cycle, preferably at the instant of glottal closure.*
   PDAs that obey this definition will be able to locate the points of glottal closure and to delimit individual laryngeal cycles. This goes beyond the scope of ordinary pitch determination in speech; if only the signal is available for the analysis, it must be totally undistorted if reliable results are to be expected. For music signals we can apply this definition if we analyze a human voice or an instrument that operates in a way similar to the human voice.
2. *T_0 is defined as the elapsed time between two successive laryngeal pulses. Measurement starts at an arbitrary point within an excitation cycle. The choice of this point depends on the individual method, but for a given PDA it is always located at the same position within the cycle.*
   Time-domain PDAs usually follow this definition. The reference point can be a significant extreme, a certain zero crossing, etc. The signal is tracked period by period in a synchronous way yielding *individual pitch periods*. This principle can be applied to both speech and music signals. In speech it may even be possible to derive the point of glottal closure from the reference point when the signal is undistorted.
3. *T_0 is defined as the elapsed time between two successive excitation cycles. Measurement starts at an arbitrary instant which is fixed according to external conditions, and ends when a complete cycle has elapsed.*
   This is an *incremental* definition. T_0 still equals the length of an individual period, but no longer from the production point of view, since the definition has nothing to do with an individual excitation cycle. The synchronous method of processing is maintained, but the phase relations between the laryngeal waveform and the markers, i. e., the pitch period delimiters at the output of the algorithm, are lost. Once a reference point in time has been established, it is kept as long as the measurement is correct and the signal remains cyclic, for instance as long as voicing continues. If this synchronization is interrupted, the reference point is lost, and the next reference point may be completely different with respect to its position within an excitation cycle.

4a. *T_0 is defined as the average length of several periods. The way in which averaging is performed, and how many periods are involved, is a matter of the individual algorithm.*
   This is the standard definition of T_0 for any PDA that applies stationary short-term analysis, including the implementations of frequency-domain PDAs. Well-known methods, such as cepstrum [10.10] or autocorrelation [10.11] approaches follow this definition. The pertinent frequency-domain definition reads as follows.

4b. *F_0 is defined as the fundamental frequency of an (approximately) harmonic pattern in the (short-term) spectral representation of the signal. It depends on the particular method in which way F_0 is calculated from this pattern.*

The perception point of view of the problem leads to a different definition of pitch [10.5]:

5. *F_0 is defined as the frequency of the sinusoid that evokes the same perceived pitch (residue pitch, virtual pitch, etc.) as the complex sound that represents the input speech signal.*
   Above all, this definition is a long-term definition [10.12]. Pitch perception theories were first developed for stationary complex sounds with constant F_0. The question of the behavior of the human ear with respect to short-term perception of time-variant pitch is not yet fully understood. The difference limen for F_0 changes, for instance, goes up by at least an order of magnitude when time-variant stimuli are involved [10.13, 14]. In practice even such PDAs that claim to be perception oriented [10.15, 16] enter the frequency domain in a similar way as in definition 4b, i. e., by some discrete Fourier transform (DFT) with previous time-domain signal windowing.

Since the results of individual algorithms differ according to the definition they follow, and since these five definitions are partly given in the time (or lag) domain and partly in the frequency domain, it is necessary to reestablish the relation between the time- and frequency-domain representations of pitch,

$$F_0 = 1/T_0 \tag{10.1}$$

in such a way that, whenever a measurement is carried out in one of these domains, however T_0 or F_0 is defined there, the representation in the other domain will always be established by this relation.

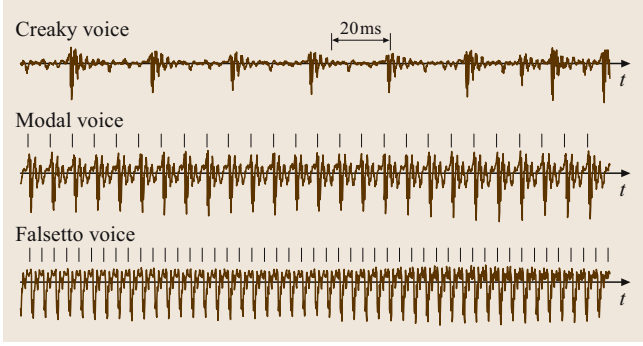


Fig. 10.2 Speech signal excitation with different voice registers (male speaker, vowel [ɛ])

10.1.2 Why is the Problem Difficult?

Literally hundreds of methods for pitch determination have been developed. None of them has been reported to be error free for any signal, application, or environmental condition.

At first glance the task appears simple: one just has to detect the fundamental frequency of a quasiperiodic signal. When dealing with speech signals, however, the assumption of (quasi)periodicity is often far from reality. For a number of reasons, the task of pitch determination must be counted among the most difficult problems in speech analysis.

- In principle, speech is a nonstationary process; the momentary position of the vocal tract may change abruptly at any time. This leads to drastic variations in the temporal structure of the signal, even between subsequent pitch periods.
- In fluent speech there are voiced segments that last only a few pitch periods [10.17].
- Due to the flexibility of articulatory gestures and the wide variety of voices, there exist a multitude of possible temporal structures. Narrowband formants at low harmonics (especially at the second or third harmonic) are a particular source of trouble.
- For an arbitrary speech signal uttered by an unknown speaker, the fundamental frequency can vary over a range of almost four octaves (50–800 Hz). Especially for female voices, F_0 thus often coincides with the first formant (the latter being about 200–1400 Hz). This causes problems when inverse-filtering techniques are applied.
- The excitation signal itself is not always regular. Even under normal conditions, i. e., when the voice is neither hoarse nor pathologic, the glottal waveform exhibits occasional irregularities. In addition, the voice may temporarily fall into vocal fry or creak ([10.18, 19]; Fig. 10.2).
- Additional problems arise in speech communication systems, where the signal is often distorted or band limited (for instance, in telephone or even mobile-phone channels).

For music signals, the situation is comparable. The range of F_0 can be even wider than for speech. However, structural changes of the signal usually occur more slowly for music. The maximum speed at which a musical instrument can be played is about 10 notes per second so that a single note usually lasts at least 100 ms. For speech, on the other hand, 100 ms is already a lot of time which

can consist of three or more segments. An additional problem in music is that we may have to analyze polyphonic signals with several pitches present at the same time.

10.1.3 Categorizing the Methods

A PDA is defined as consisting of three processing steps: (a) the preprocessor, (b) the basic extractor, and (c) the postprocessor [10.1, 20]. The basic extractor performs the main task of converting the input signal into a series of pitch estimates. The task of the preprocessor is data reduction and enhancement in order to facilitate the operation of the basic extractor. The postprocessor (Sect. 10.5.4) is more application oriented. Typical tasks are error correction, pitch tracking, and contour smoothing, or visualization.

The existing PDA principles can be split into two gross categories when the input signal of the basic extractor is taken as a criterion. If this signal has the same time base as the original input signal, the PDA operates in the time domain. It will thus measure T_0 according to one of the definitions 1–3 above. In all other cases, somewhere in the preprocessor the time domain is left. Since the input signal is time variant, this is done by a short-term transform; and we will usually determine T_0 or F_0 according to definitions 4a,b or 5; in some cases definition 3 may apply as well. Accordingly, we have the two categories: time-domain PDAs, and short-term analysis PDAs. These will be discussed in the next two sections.

10.2 Short-Term Analysis PDAs

In any short-term analysis PDA a short-term (or short-time) transformation is performed in the preprocessor. The speech signal is split into a series of frames; an individual frame is obtained by taking a limited number of consecutive samples of the signal $s(n)$ from a starting point, $n = q$, to the ending point, $n = q + K$. The frame length K (or $K + 1$) is chosen short enough so that the parameter(s) to be measured can be assumed approximately constant within the frame. On the other hand, K must be large enough to guarantee that the parameter remains measurable. For most short-term analysis PDAs a frame thus requires two or three complete periods at least. In extreme cases, when F_0 changes abruptly, or when the signal is irregular, these two conditions are in conflict with each other and may become a source of error [10.21]. The frame interval Q, i. e., the distance between consecutive frames (or its reciprocal, the frame rate), is determined in such a way that any significant parameter change is documented in the measurements. 100 frames/s, i. e., $Q = 10$ ms, is a usual value.

The short-term transform can be thought of as behaving like a concave mirror that focuses all the information on pitch scattered across the frame into one single peak in the spectral domain. This peak is then determined by a peak detector (the usual implementation of the basic extractor in this type of PDAs). Hence this algorithm yields a sequence of average pitch estimates. The short-term transform causes the phase relations between the spectral domain and the original signal to be lost. At the same time, however, the algorithm loses much of its sensitivity to phase distortions and signal degradation.

Not all the known spectral transforms show the desired focusing effect. Those that do are in some way related to the (power) spectrum: correlation techniques, frequency-domain analysis, active modeling, and statistical approaches (Fig. 10.3). These methods will be discussed in more detail in the following.

10.2.1 Correlation and Distance Function

Among the correlation techniques we find the well-known short-term autocorrelation function (ACF)

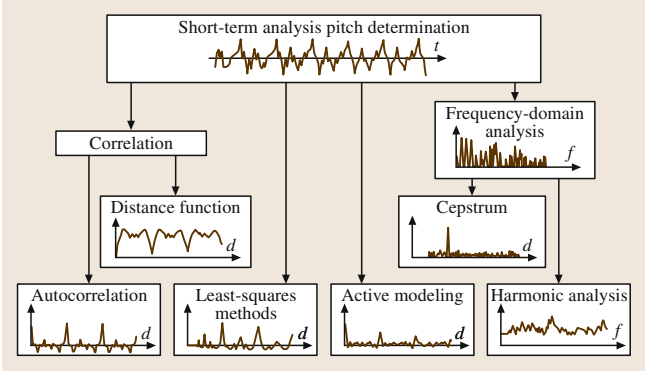


Fig. 10.3 Methods of short-term analysis (short-time analysis) pitch determination. (Time t and lag d scales are identical; the frequency f scale in the box 'Harmonic analysis' has been magnified)

usually given by

$$r(d,q) = \sum_{n=q}^{q+K-d} s(n)s(n+d)\,. \tag{10.2}$$

The autocorrelation function of a periodic signal exhibits a strong peak when the lag d equals the period T_0/T of the signal, T being the time-domain sampling interval of the signal.

The autocorrelation PDA is among the oldest principles for short-term analysis PDAs. However, it tends to fail when the signal has a strong formant at the second or third harmonic. Therefore this technique became successful in pitch determination of band-limited speech signals when it was combined with time-domain linear or nonlinear preprocessing, such as center clipping or inverse filtering [10.22, 23].

The counterpart to autocorrelation is given by applying a distance function, for instance the average magnitude difference function (AMDF) [10.24, 25]:

$$\mathrm{AMDF}(d,q) = \sum_{n=q}^{q+K} |s(n) - s(n+d)|\,. \tag{10.3}$$

If the signal were strictly periodic, the distance function would vanish at the lag (delay time) $d = T_0/T$. For quasiperiodic signals there will be at least a strong minimum at this value of d. So, in contrast to other short-term PDAs where the estimate of T_0 or F_0 is indicated by a maximum whose position and value have to be determined, the minimum has an ideal target value of zero so that we only need to determine its position. For this reason, distance functions do not require (quasi)-stationarity within the measuring interval; they can cope with very short frames of one pitch period or even less. This principle is thus able to follow definition 3.

Shimamura and *Kobayashi* [10.26] combine ACF and AMDF in that they weight the short-term ACF with the reciprocal of the AMDF, thus enhancing the principal peak of the ACF at $d = T_0/T$. For the PDA they named YIN, *De Cheveigné* and *Kawahara* [10.27] start from a squared distance function,

$$D(d,q) = \sum_{n=q}^{q+K} [s(n) - s(n+d)]^2 \tag{10.4}$$

and normalize it to increase its values at low lags,

$$D'(d,q) = \frac{D(d,q)}{\frac{1}{d}\sum_{\delta=1}^{d} D(\delta,q)}\;;\; d>0 \tag{10.5}$$

with $D'(0) = 1$. In doing so, the authors were able to drop the high-frequency limit of the measuring range and to apply their PDA to high-pitched music signals as well. The normalized distance function is locally interpolated around its minima to increase the accuracy of the value of D' at the minima and the pitch estimate at the same time.

Knowing that many errors arise from a mismatch during short-term analysis (which results in too few or too many pitch periods within a given frame), *Fujisaki* et al. [10.21] investigated the influence of the relations between the error rate, the frame length, and the actual value of T_0 for an autocorrelation PDA that operates on the linear prediction residual. The optimum occurs when the frame contains about three pitch periods. Since this value is different for every individual voice, a fixed-frame PDA runs nonoptimally for most situations. For an exponential window, however, this optimum converges to a time constant of about 10 ms for all voices. For a number of PDAs, especially for the autocorrelation PDA, such a window permits recursive updating of the autocorrelation function, i. e., sample-by-sample pitch estimation without excessive computational effort.

Hirose et al. [10.28] and *Talkin* [10.17] showed that the autocorrelation function can also be computed in a nonstationary way using a suitable normalization,

$$r(d,q) = \frac{\sum_{n=q}^{q+K} s(n)s(n+d)}{\sqrt{\left[\sum_{n=q}^{q+K} s^2(n)\right]\left[\sum_{n=q}^{q+K} s^2(n+d)\right]}}\,. \tag{10.6}$$

In Talkin's PDA a 7.5 ms frame is used; the effective frame length is of course 7.5 ms plus the longest pitch period in the measuring range.

Terez [10.29] applies a multidimensional embedding function and a scatter plot procedure derived from chaos theory. The underlying idea, however, is quite straightforward and leads to a distance function in a multidimensional state space. The problem is how to convert the one-dimensional speech signal into a multidimensional representation. In Terez's algorithm a vector is formed from several equally spaced samples of the signal,

$$\boldsymbol{s}(n) = \left[s(n)\; s(n+d)\; \cdots\; s(n+Nd)\right]^{\mathrm{T}}\,, \tag{10.7}$$

(where the frame reference point q has been omitted here and in the following for sake of simplicity) whose components create an N-dimensional space, the state space. In Terez's algorithm, $N = 3$ and $d = 12$ samples gave the best results. If the signal is voiced, i. e., cyclic, the vector $\boldsymbol{s}$ will describe a closed curve in the state space as time proceeds, and after one pitch period it is

expected to come back near the starting point. We can thus expect the (Euclidian) distance

$$D(n,\,p) = \|\boldsymbol{s}(n) - \boldsymbol{s}(n+p)\| \tag{10.8}$$

generally to become a minimum when the trial period p equals the true period T_0/T. If we compute $D(n,\,p)$ for all samples $s(n)$ within the frame and all values of p within the measuring range and count the number of events, depending on p, where D lies below a predetermined threshold, we arrive at a periodicity histogram that shows a sharp maximum at $p = T_0/T$.

As it develops the distance function D for all samples of a frame, this PDA follows the short-term analysis principle. Yet one can think of running it with a comparatively short window, thus following definition 3.

The idea of using a multidimensional representation of the signal for a PDA (and VDA) dates back to the 1950s [10.1]. In 1964 *Rader* [10.30] published the *vector* PDA where he used the output signals from a filterbank (cf. *Yaggi* [10.31], Sect. 10.3.3) and their Hilbert transforms to form a multidimensional vector $\boldsymbol{s}(n, q)$. Rader then used the Euclidian distance between the vector at the starting point $n = q$ of the measurement and the points $q + p$ to set up a distance function which shows a strong minimum when p equals the true period T_0/T. This PDA follows definition 3 as well.

Medan et al. [10.32] present a PDA (they called the super-resolution PDA) that explicitly addresses the problem of granularity due to signal sampling and applies a short-term window whose length depends on the trial pitch period p in that it takes on a length of exactly $2p$. A similarity function is formulated that expresses the relation between the two periods in the window,

$$\begin{aligned} \boldsymbol{s}(n, q) &= a \cdot \boldsymbol{s}(n + p, q) + e(n, q)\,; \\ & n = q, \cdots, q + p - 1\,. \end{aligned} \tag{10.9}$$

Here, a is a positive amplitude factor that takes into account possible intensity changes between adjacent periods. Equation (10.9) is optimized with respect to a and the unknown period p applying a least-squares criterion to minimize the error e,

$$\hat{p} = \operatorname{argmin}_{a,e} \left(\frac{\sum_{n=q}^{q+p-1} [s(n) - as(n+p)]^2}{\sum_{n=q}^{q+p-1} s^2(n)} \right)\,. \tag{10.10}$$

This optimization finally results in maximization of the correlation term

$$\hat{p} = \operatorname{argmax} \left(\frac{\left[\sum_{n=q}^{q+p-1} s(n)s(n+p) \right]^2}{\left[\sum_{n=q}^{q+p-1} s^2(n) \right] \left[\sum_{n=q}^{q+p-1} s^2(n+p) \right]} \right)\,. \tag{10.11}$$

This resembles Talkin's ACF approach [10.17] except that here the trial period p determines the length of the window as well.

From (10.11) a pitch period estimate can only be derived as an integer number of samples. In a second pass, this estimate is refined (to yield the super-resolution) by expanding (10.11) for a fraction of a sample using linear interpolation.

10.2.2 Cepstrum and Other Double-Transform Methods

The sensitivity against strong first formants, especially when they coincide with the second or third harmonic, is one of the big problems in pitch determination. This problem is suitably met by some procedure for spectral flattening.

Spectral flattening can be achieved in several ways. One of them is time-domain nonlinear distortion, such as center clipping ([10.11, 22]; see previous section). A second way is linear spectral distortion by inverse filtering (e.g., [10.23]). A third way is frequency-domain amplitude compression by nonlinear distortion of the spectrum. This algorithm operates as follows: (1) short-term analysis and transformation into the frequency domain via a suitable discrete Fourier transform (DFT), (2) nonlinear distortion in the frequency domain, and (3) inverse DFT back into the time domain (which we will call the lag domain to avoid ambiguity).

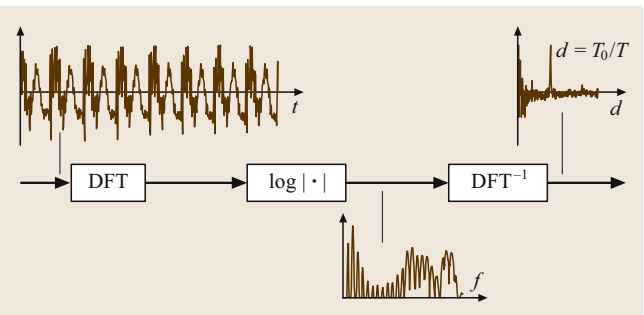


Fig. 10.4 Cepstrum pitch determination. Signal: vowel [i], 48 ms

If we take the logarithm of the power spectrum as the frequency-domain nonlinear distortion, we arrive at the well-known cepstrum PDA ([10.10]; Fig. 10.4). Instead of the logarithmic spectrum, however, one can also compute the amplitude spectrum or its square root and transform it back [10.33, 34]. The inverse Fourier transform of the power spectrum gives the autocorrelation function. All these so-called double-transform techniques [10.33] lead to a lag-domain representation that exhibits a strong maximum at the lag $d = T_0/T$. The independent variable, lag (or *quefrency*, as it is called with respect to the cepstrum [10.10]), has the physical dimension of time, but as long as the phase relations are discarded by the nonlinear distortion, all values of d refer to a virtual point $d = 0$ where we will always find a pitch pulse, and then the next one necessarily shows up at $d = T_0/T$.

Two members of this group were already mentioned: the autocorrelation PDA [10.11] and the cepstrum PDA [10.10]. It is well known that the autocorrelation function can be computed as the inverse Fourier transform of the power spectrum. Here, the distortion consists of taking the squared magnitude of the complex spectrum. The cepstrum, on the other hand, uses the logarithm of the spectrum. The two methods therefore differ only in the characteristics of the respective nonlinear distortions applied in the spectral domain. The cepstrum PDA is known to be rather insensitive to strong formants at higher harmonics but to develop a certain sensitivity to additive noise. The autocorrelation PDA, on the other hand, is insensitive to noise but rather sensitive to strong formants. Regarding the slope of the distortion characteristic, we observe the dynamic range of the spectrum being expanded by squaring the spectrum for the autocorrelation PDA, whereas the spectrum is substantially flattened by taking the logarithm. The two requirements – robustness against strong formants and robustness against additive (white) noise – are in some way contradictory. Expanding the dynamic range of the spectrum emphasizes strong individual components, such as formants, and suppresses wideband noise, whereas spectral flattening equalizes strong components and at the same time raises the level of low-energy regions in the spectrum, thus raising the level of the noise as well. Thus it is worth looking for other characteristics that perform spectral amplitude compression. *Sreenivas* [10.36] proposed the fourth root of the power spectrum instead of the logarithm. For larger amplitudes this characteristic behaves very much like the logarithm; for small amplitudes, however, it has the advantage of going to zero and not to $-\infty$. *Weiss* et al. [10.33] used the amplitude spectrum, i. e., the magnitude of the complex spectrum.

10.2.3 Frequency-Domain Methods: Harmonic Analysis

Direct determination of F_0 as the location of the lowest peak in the power or amplitude spectrum is unreliable and inaccurate; it is preferable to investigate the harmonic structure of the signal so that all harmonics contribute to the estimate of F_0. One way to do this is spectral compression combined with harmonic pattern matching, which computes the fundamental frequency as the greatest common divider of all harmonics. The power spectrum is compressed along the frequency axis by a factor of two, three etc. and then added to the original power spectrum. This operation gives a peak at F_0 resulting from the coherent additive contribution of the higher harmonics [10.35, 37]. Some of these PDAs stem from theories and functional models of pitch perception in the human ear [10.12, 15, 16].

The PDA described by *Martin* [10.35] (Fig. 10.5) modifies the harmonic pattern-matching principle in such a way that the computational effort for the spectral transform is minimized. The signal is first decimated to 4 kHz and then Fourier transformed by a 128 point fast Fourier transform (FFT). This yields a spectral resolution of about 30 Hz, which is sufficient to represent

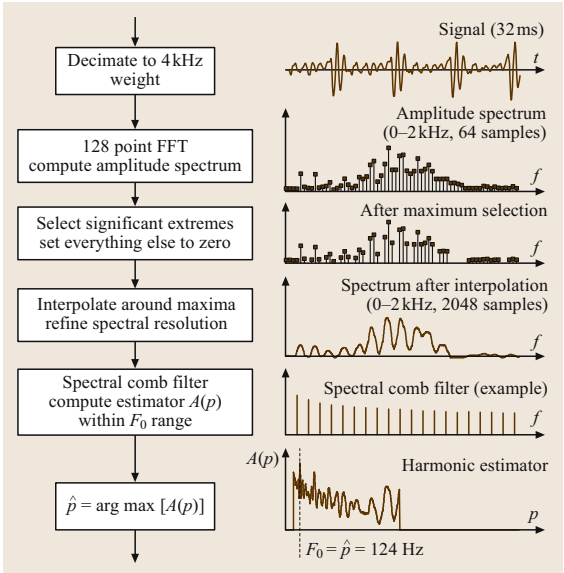


Fig. 10.5 Frequency-domain PDA by harmonic compression and pattern matching (after *Martin* [10.35])

values of F_0 down to 60 Hz. The algorithm then enhances any spectral information that may pertain to a harmonic structure by preserving only those spectral samples that represent local maxima or their immediate neighbors and setting everything else to zero. To measure F_0 with sufficient accuracy, the spectral resolution is then increased to 1 Hz. A spectral comb filter, which is applied over the whole range of F_0, yields the harmonic estimation function $A(p)$; the value of p where this function reaches its maximum is taken as the estimate for F_0. In a more-recent version of this PDA, *Martin* [10.38, 39] applies a logarithmic frequency scale for the computation of $A(p)$, which results in another substantial reduction of the computational effort and has the additional advantage that the relative accuracy of the PDA is now constant over the whole range of F_0.

Similar to *Martin*'s [10.38] PDA for speech, *Brown* [10.41] developed a frequency-domain PDA for music which uses a logarithmic frequency scale. In Brown's PDA the spacing on the frequency axis equals a quarter tone (about 3%), i. e., 1/24 of an octave. In such a scale that corresponds to a musical interval scale, a harmonic structure, if the timbre is unchanged, always takes on the same pattern regardless of the value of its fundamental frequency (Fig. 10.6). Consequently, a pattern-recognition algorithm is applied to detect such patterns in the spectrum and to locate their starting point corresponding to F_0. The patterns themselves depend on the kind of instrument analyzed and can be adjusted according to the respective instruments.

Special attention is given to the frequency resolution of the PDA. To apply a pattern-recognition method, patterns are expected to align with the semitone scale. This requires the frequency scale spacing of a quarter tone. To make sure that a harmonic shows up in one and only one frequency bin, we need a window length of 34 fundamental periods to satisfy the sampling theorem in the frequency domain. For $F_0 = 100$ Hz this would give a window of 340 ms, which is far beyond reality for speech and even excessive for music. However, if we adapt the window-length requirement to the fundamental frequency to be determined, we would need 34 periods at any F_0 to be measured, which results in much shorter windows for higher-frequency bins. For the DFT, this leads to a window whose length is inversely proportional to the bin's frequency. If the spectral values are computed individually, both an individual time-domain window for each frequency bin and unequal spacing of the frequency bins are possible. *Brown* and *Puckette* [10.40] showed that a fast Fourier transform can be applied if its kernel is slightly modified. The PDA by *Medan* et al. [10.32] is a time-domain counterpart to this approach.

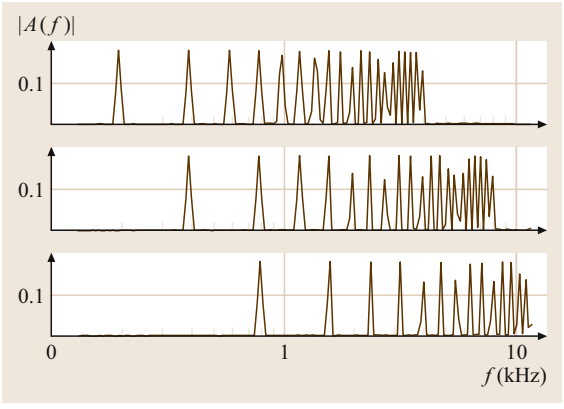


Fig. 10.6 Harmonic patterns in log frequency scale (after *Brown* and *Puckette* [10.40])

As the accuracy of this PDA was not sufficient to determine F_0 for instruments that can vary their frequencies continuously (such as string or wind instruments or a human voice), and as the required window length was excessive even for music, a PDA with a 25 ms window was developed [10.40] whose frequency resolution was refined using a phase-change technique. This technique is based on the instantaneous-frequency approach by *Charpentier* [10.42] (see below) who used the short-term phase spectrum at two successive samples in time to determine the frequencies of the individual harmonics without needing spectral interpolation. When a Hann window is used to weight the time signal, the time shift of one sample can be recursively computed in the frequency domain without needing another DFT.

Lahat et al. [10.43] transfer the autocorrelation method into the frequency domain. The amplitude spectrum is passed through a bank of 16 spectral filters (*lifters*), which cover the measuring range of F_0. At the output of each lifter a frequency domain autocorrelation function is calculated covering the respective range of each lifter. The estimate for F_0 is then determined as the location of the maximum of that function and refined by interpolation.

For harmonic analysis it is often convenient to estimate the number of harmonics, i. e., the order of a harmonic model, simultaneously with the fundamental frequency. For instance, *Doval* and *Rodet* [10.44] apply such a procedure for a PDA with an extremely wide measuring range (40–4000 Hz) for music signals. The algorithm is based on a harmonic-matching procedure using a maximum-likelihood criterion. To obtain

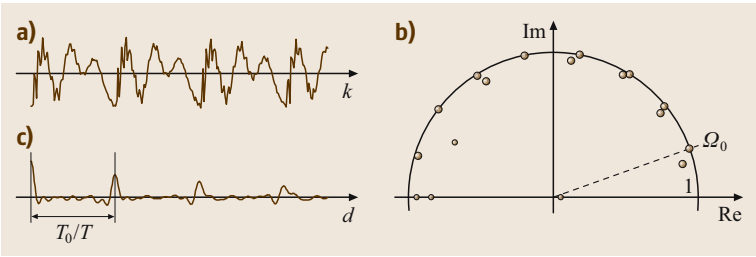


Fig. 10.7a–c PDA with active modeling. **(a)** Signal: 32 ms, vowel [e], male voice; **(b)** zeros of the 41-st-order LP polynomial in the z plane (upper half; sampling frequency reduced to 2 kHz); **(c)** reconstructed impulse response with zero phase and equal amplitude of all partials

a rapid initial estimate, the measuring range is split into subranges that have an equal number of partials in the frequency range for which the spectrum is analyzed. The initial estimate is obtained by selecting the subrange that is most likely to contain the optimal estimate for F_0. For the final step of refining the estimate, only this subrange is further processed.

The principle of instantaneous frequency (IF) was introduced into pitch determination by *Charpentier* [10.42]. Instantaneous frequency is defined as the time derivative of the instantaneous phase,

$$\dot{\varphi}(t) := \frac{\mathrm{d}\varphi}{\mathrm{d}t} \quad \text{for } s(t) = a(t)\exp[\mathrm{i}\omega(t)t]\,, \tag{10.12}$$

where $a(t)$ is the instantaneous amplitude. The short-term Fourier transform can be viewed as a set of bandpass filters as follows,

$$S(f,t) = \int_{-\infty}^{\infty} s(\tau)w(t-\tau)\mathrm{e}^{-2\pi\mathrm{i}f\tau}\,\mathrm{d}\tau \tag{10.13}$$

$$= \mathrm{e}^{-2\pi\mathrm{i}ft}\int_{-\infty}^{\infty} s(\tau)w(t-\tau)\mathrm{e}^{-2\pi\mathrm{i}f(t-\tau)}\,\mathrm{d}\tau$$

$$F(f,t) = \mathrm{e}^{2\pi\mathrm{i}ft}S(f,t)\,. \tag{10.14}$$

Here, the signal F is the output of the *bandpass* centered around the frequency f. The IF for this signal becomes

$$\dot{\varphi}(f,t) = \frac{\partial}{\partial t}\arg[F(f,t)]\,. \tag{10.15}$$

There are different ways to effectively compute the IF from the discrete short-term spectrum [10.42,45,46]. The bandpass filters have a certain bandwidth that depends on the time-domain window applied [10.42] and extends over more than one DFT coefficient. If we now compute the IF for each frequency bin of the DFT spectrum of a voiced speech signal, the IFs of bins adjacent to a strong harmonic tend to cluster around the true frequency of this harmonic, and so it is possible to enhance the harmonics in the spectrum if the bins are regrouped according to their respective IFs, thus forming the so-called IF amplitude spectrum. *Abe* et al. [10.45] transform the IF amplitude spectrum back into the time domain, thus obtaining a representation similar to that of a double-spectral-transform PDA. *Nakatani* and *Irino* [10.46] define a spectral dominance function that suppresses insignificant information in the IF amplitude spectrum and derive F_0 by harmonic matching of this dominance function.

10.2.4 Active Modeling

Linear prediction (LP) is usually applied to estimating the transfer function of the vocal tract. If a high-order LP filter is applied to a voiced speech signal, however, its poles will match the individual harmonics of the signal rather than the resonances of the vocal tract. A PDA based on this principle was designed by *Atal* (unpublished; see [10.47], or [10.1]). The algorithm operates as follows (Fig. 10.7):

- After decimation to 2 kHz, the signal is analyzed with a 41-st-order LP filter using the covariance method of linear prediction. The high order guarantees that even at the low end of the F_0 range, i. e., at $F_0 = 50$ Hz, two complex poles are available for each harmonic. Each complex pole pair represents an exponentially decaying (or rising) oscillation.
- To eliminate phase information, all residues at the pole locations in the z plane are set to unity. The pertinent computation can be avoided when the locations of the poles are explicitly determined.
- The impulse response of the filter now supplies a waveform for the estimation of T_0 (Fig. 10.7c). When the poles are explicitly available, it is sufficient to determine and to sum up the impulse responses of the individual second-order partial filters. This method has the advantage that the sampling frequency of the impulse response – and with it the measurement accuracy – can easily be increased. In addition, poles that are situated outside or far inside the unit circle can be modified or excluded from further processing.

Arévalo [10.48] showed that this PDA is extremely robust to noise and that one can also use it with short frame lengths so that it matches definition 3.

10.2.5 Least Squares and Other Statistical Methods

The first statistical approach in pitch determination is based on a least-squares principle. Originally this approach was based on a mathematical procedure to separate a periodic signal of unknown period T_0 from Gaussian noise within a finite signal window [10.49]. It computes the energy of the periodic component at a trial period τ and varies τ over the whole range of T_0. The value of τ that maximizes the energy of the periodic component for the given signal is then taken as the estimate of T_0. *Friedman* [10.50] showed that this PDA has a trivial maximum when τ equals the window length K, and developed a work-around.

With respect to robustness, the least-squares PDA behaves like the autocorrelation principle: it is extremely robust against noise but somewhat sensitive to strong formants. However, there is no algorithmic shortcut so that an order of K^2 operations are needed to compute the estimate for a frame. So this PDA was slower than its counterparts that can make use of the FFT; hence this principle was not further pursued until more powerful computers became available.

The method was revived with the upcoming of the sinusoidal model of speech [10.2]. The continuous speech signal $s(n)$ is modeled as a sum of sinusoids with time-varying amplitudes, frequencies, and phases. Within a short-term frame, these parameters can be assumed constant,

$$s(n) = \sum_{m=1}^{M} S_m \ \exp\left(\mathrm{i}\Omega_m n + \varphi_m\right) \ . \tag{10.16}$$

The parameters of this model are estimated from the peaks within the short-term Fourier spectrum of the frame. This can be converted into a PDA [10.51] when the sinusoidal representation in (10.16), whose frequencies are generally *not* harmonics of a fundamental, is matched against a harmonic model,

$$u(n) = \sum_{k=1}^{K} U_k \ \exp\left(\mathrm{i}k\Omega_0 n + \psi_k\right) \ . \tag{10.17}$$

Starting from the difference between $s(n)$ and $u(n)$, the match is done using a modified least-squares criterion, which finally results in maximizing the expression with respect to the trial (angular) fundamental frequency p,

$$\varrho(p) = \sum_{k=1}^{K(p)} U(kp) \times \left\{ \max_{\Omega_m \in L(kp)} \left[S_m D(\Omega_m - kp)\right] - \frac{1}{2} U(kp) \right\} \ . \tag{10.18}$$

Like the model of virtual-pitch perception [10.4], this criterion takes into account near-coincidences between a harmonic kp and the (angular) frequency Ω_m of the respective component of the sinusoidal model, and it defines a lobe L of width p around each harmonic and the corresponding weighting function

$$D(\Omega - kp) = \frac{\sin[2\pi(\Omega - kp)/p\,]}{(\Omega - kp)/p} \tag{10.19}$$

within the lobe, and zero outside. The lobe becomes narrow for low values of p and broader for higher values. If there are several components Ω_m within a lobe, only the largest is taken. The amplitude estimates $U(kp)$ are derived from a smoothed Fourier or LP spectrum of the frame. The measurement may be confined to a subband of the signal, e.g., to 2 kHz.

Both the sinusoidal model and the PDA have been applied to speech and music signals.

For pitch detection in noisy speech, third-order statistics are occasionally applied. One such PDA was developed by *Moreno* and *Fonollosa* [10.52]. Their PDA applies a special third-order cumulant,

$$C(0, d) := \sum_{n} s(n)\, s(n)\, s(n+d) \ , \tag{10.20}$$

which tends to vanish for noises with symmetrical distribution, such as Gaussian noise. It also tends to vanish for voiceless fricatives, as *Wells* [10.53] discovered for his VDA. If the signal is periodic, the cumulant C is also periodic, but one cannot expect a maximum to occur at $d = 0$. This PDA thus treats $C(0, d)$ like an ordinary signal, takes the autocorrelation function, and derives T_0 therefrom in the same way as it is done in an ordinary autocorrelation PDA. The algorithm was tested with speech in various additive noises at various signal-to-noise ratios (SNRs) against an autocorrelation PDA (without any pre- or postprocessing) and found superior, especially when noise levels were high.

The PDA by *Tabrikian* et al. [10.54] determines the parameters of a harmonic-plus-noise model by maximizing a log-likelihood measure, i. e., the unknown fundamental frequency, the spectral amplitudes of the

harmonics, and the variance of the noise, which is modeled to be Gaussian. It then performs pitch tracking over consecutive frames using a method based on the maximum a posteriori probability. The authors tested their PDA under extreme noise conditions and found that it worked even at SNRs of −5 dB and worse. A similar algorithm was developed by *Godsill* and *Davy* [10.55] for music signals.

10.2.6 Concluding Remarks

Short-term analysis PDAs provide a sequence of average pitch estimates rather than a measurement of individual periods. They are not very sensitive to phase distortions or to the absence of the first partial. They collect information about pitch from many features and (mostly) from several periods of the signal. They are thus robust against additive noise. Some of them still work at SNRs of 0 dB or worse. On the other hand, they are sensitive when the signal does not fulfil their basic requirement, i. e., periodicity. Rapid within-frame changes of F_0 of irregularly excited signals (e.g., laryngealizations) lead to failure.

One advantage of this principle that is not always explicitly mentioned is the ability to give rather accurate estimates and to overcome measurement granularity due to signal sampling. To decrease computational complexity, many of these PDAs perform some moderate low-pass filtering and/or decrease the sampling frequency in the first step and thus increase the granularity. Once a crude estimate is available, it can be refined via a local interpolation routine, which is frequently implemented. This is most evident in active modeling (Sect. 10.2.4) where the impulse response of the model filter can be generated with an arbitrarily high sampling frequency independently from the sampling frequency of the signal. However, any other representation from which pitch is derived – ACF, AMDF, cepstrum, etc. – can be treated like a signal and can be locally upsampled, e.g., via a standard multirate finite impulse response (FIR) filter, to increase measurement accuracy. The evaluation by *McGonegal* et al. [10.56] showed that an increased accuracy is honored by the human ear when listening to synthetic speech generated with such a pitch contour.

10.3 Selected Time-Domain Methods

This category of PDAs is less homogenous than that of the short-term analysis methods. One possibility to group them is according to how the data reduction is distributed between the preprocessor and the basic extractor, and we find most of these PDAs between two extremes.

- Data reduction is done in the preprocessor. In the extreme case, only the waveform of the first harmonic is offered to the basic extractor. The basic extractor processes this harmonic and derives pitch from it.
- Data reduction is done in the basic extractor, which then has to cope with the whole complexity of the temporal signal structure. In the extreme case, the preprocessor is totally omitted. The basic extractor investigates the temporal structure of the signal, extracts some key features, and derives the information on pitch therefrom.

A third principle is situated somewhere in the middle of these extremes. Temporal structure simplification performs a moderate data reduction in the preprocessor but preserves the harmonic structure of the signal.

Time-domain PDAs are principally able to track the signal period by period. At the output of the basic extractor we usually obtain a sequence of period boundaries (pitch *markers*). Since the local information on pitch is taken from each period individually, time-domain PDAs are sensitive to local signal degradations and are thus less reliable than most of their short-term analysis counterparts. On the other hand, time-domain PDAs may still operate correctly even when the signal itself is aperiodic (but still cyclic), in speech for instance due to temporary voice perturbation or laryngealization.

Most time-domain PDAs, especially those which follow definitions 2 and 3, were developed before the 1990s. With the introduction of time-domain pitch modification methods [10.57], research in this area concentrated on high-precision algorithms for determination of the instant of glottal closure. This issue will be discussed in Sect. 10.7.1.

10.3.1 Temporal Structure Investigation

A pitch period in speech is the truncated response of the vocal tract to an individual glottal impulse. Since the vocal tract behaves like a lossy linear system, its impulse response consists of a sum of exponentially damped oscillations. It is therefore to be expected that

the magnitude of the significant peaks in the signal is greater at the beginning of the period than versus the end. Appropriate investigation of the signal peaks (maxima and/or minima) leads to an indication of periodicity.

There are some problems with this approach, however. First, the frequencies of the dominant damped waveforms are determined by the local formant pattern and may change abruptly. Second, damping of the formants, particularly of a low first formant, is often quite weak and can be hidden by temporary changes of the signal level due to articulation. Third, if the signal is phase distorted, different formants may be excited at different points in time. These problems are solvable but lead to complex algorithmic solutions investigating a great variety of temporal structures.

The usual way to carry out the analysis is the following [10.58].

- Do a moderate low-pass filtering to remove the influence of higher formants.
- Determine all the local maxima and minima in the signal.
- Exclude those extremes that are found to be insignificant until one significant point per period is left; this point will become the local pitch marker.
- Reject obviously wrong markers by local correction.

Many individual (and heuristic) solutions have been developed, but they cannot be reviewed here for lack of space. For more details, the reader is referred to the literature [10.1].

10.3.2 Fundamental Harmonic Processing

F_0 can be detected in the signal via the waveform of the fundamental harmonic. If present in the signal, this harmonic is extracted from the signal by extensive low-pass filtering in the preprocessor. The basic extractor can then be relatively simple. Figure 10.8 shows the principle of three basic extractors: zero-crossings analysis as the simplest one, nonzero threshold analysis, and finally threshold analysis with hysteresis. The zero-crossings analysis basic extractor sets a marker whenever the zero axis is crossed with a defined polarity. This requires that the input waveform has two and only two zero crossings per period. The threshold analysis basic extractor sets a marker whenever a given nonzero threshold is exceeded. When operating with hysteresis, the marker is only set when a second (lower) threshold is crossed in the opposite direction. This more-elaborate device requires less low-pass filtering in the preprocessor.

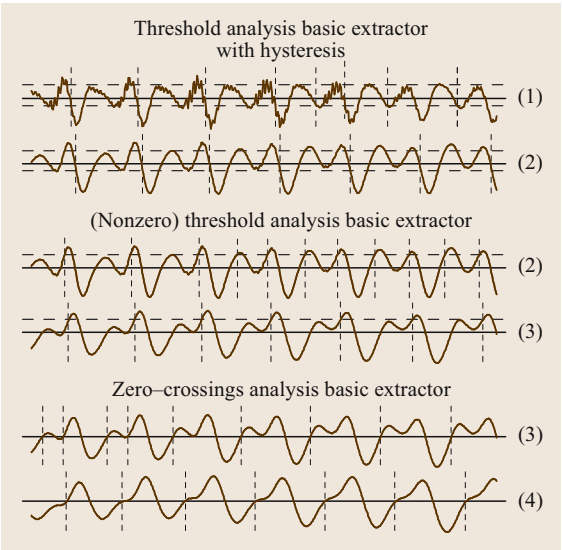


Fig. 10.8 Example of the performance of basic extractors for fundamental harmonic extraction in speech. Signals: (1) original (vowel [i], 32 ms), (2) low-pass filtered at 6 dB/oct, (3) low-pass filtered at 12 dB/oct, and (4) low-pass filtered at 18 dB/oct. The signal is such that success and failure are displayed at the same time

The requirement for extensive low-pass filtering is a severe weak point of this otherwise fast and simple principle when applied to speech signals. In a number of applications, however, such as voice quality measurement or the preparation of reference elements for time-domain speech synthesis, where the signals are expected to be clean, the use of a PDA applying first-partial processing may be advantageous. *Dologlou* and *Carayannis* [10.59] proposed a PDA that overcomes a great deal of the problems associated with the low-pass filter. An adaptive linear-phase low-pass filter that consists of a variable-length cascade of second-order filters with a double zero in the z plane at $z = -1$ is applied. These filters are consecutively applied to the input signal; after each iteration the algorithm tests whether the higher harmonics are sufficiently attenuated; if they are, the filter stops. T_0 is then derived from the remaining first partial by a simple maximum detector. Very low-frequency noise is tolerable since it barely influences the positions of the maxima.

10.3.3 Temporal Structure Simplification

Algorithms of this type take some intermediate position between the principles of structural analysis and

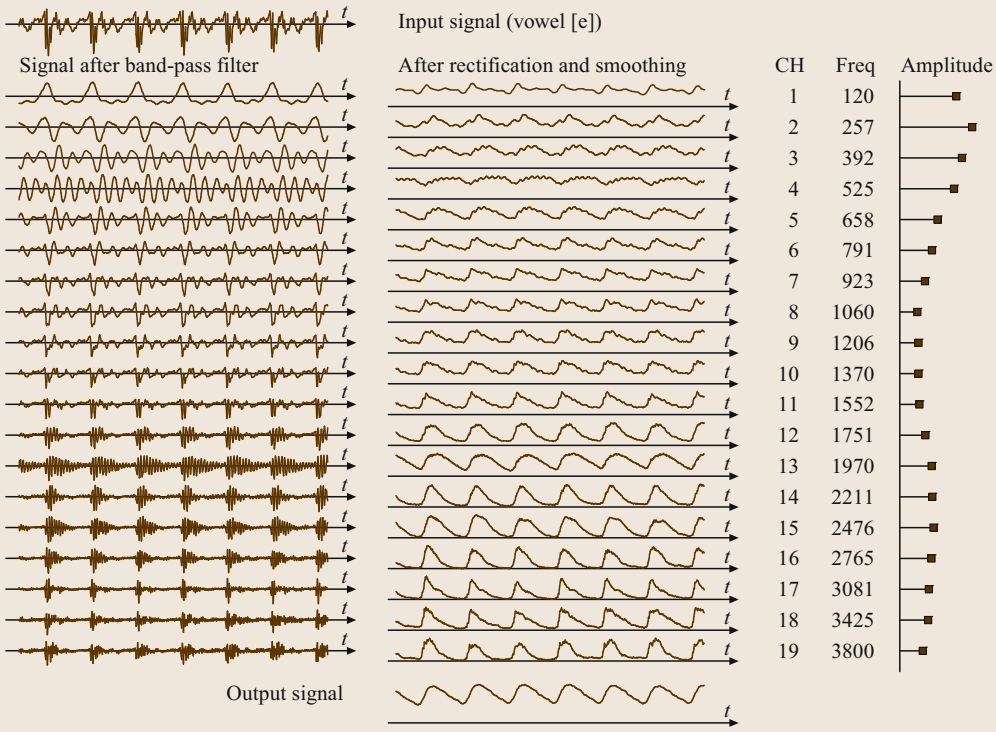


Fig. 10.9 Example PDA by *Yaggi* (1962 [10.31]). The signal is split into 19 subbands. In each channel (CH) the filtered signal is rectified and smoothed; the weighted outputs of the channels are added, and pitch markers are derived from the resulting signal via maximum detection

fundamental harmonic extraction. The majority of these algorithms follow one of two principles: (1) inverse filtering, and (2) epoch detection. Both of these principles deal with the fact that the laryngeal excitation function has a temporal structure that is much simpler and more regular than the temporal structure of the speech signal itself, and both methods when they work properly are able to follow definition 1 if the signal is not grossly phase distorted.

The inverse filter approach cancels the transfer function of the vocal tract and thus reconstructs the laryngeal excitation function. If one is interested in pitch only and not in the excitation function itself, a crude approximation of the inverse filter is sufficient. For instance, we can confine the analysis to the first formant [10.60].

The second principle, epoch extraction [10.61], is based on the fact that at the beginning of each laryngeal pulse there is a discontinuity in the second derivative of the excitation function. Usually this discontinuity cannot be reliably detected in the speech signal because of phase distortions that occur when the waveform passes the vocal tract. The signal is thus first phase shifted by 90° (by applying a Hilbert transform). The squares of the original and the phase-shifted signals are then added to yield a new signal that shows a distinct peak at the time when the discontinuity in the excitation function occurs. In principle this yields the instantaneous amplitude of the complex analytic signal constructed from the original signal as its real part and the phase-shifted signal as its imaginary part.

The original method [10.61] works best when the spectrum of the investigated signal is flat to some extent. To enforce spectral flatness, the analyzed signal can be band limited to high frequencies well above the narrowband lower formants. Another way is to analyze the LP residual or to filter the signal into subbands.

One prototype of these algorithms, which never became widely known, was developed by *Yaggi* [10.31]. It splits the signal into 19 subbands and subsequently rectifies and smoothes the signal in each channel so that the envelope is extracted. The individual channels are then added, and the individual periods are derived from

the resulting signal (Fig. 10.9). Another prototype is the PDA by *Dolanský* [10.62] that models the envelope of the pitch period by a decaying exponential function (in analog technology) and sets a marker whenever the signal exceeds the modeled envelope, resetting the envelope at the same time.

Both the inverse filter approach and the epoch detection principle have one weak point, which frequently arises with female voices. When F_0 is high, it may coincide with the first formant $F1$. In this case the signal becomes nearly sinusoidal since we have something like a double pole *glottal formant* and $F1$ at the same frequency) in the overall transfer function. If an inverse filter is not blocked in this case, it removes the fundamental harmonic from the signal and brings the PDA to failure. For epoch detection, we know that the envelope of a sinusoid is a constant ($\cos^2 x + \sin^2 x = 1$) and does not show any discontinuity. Hence these algorithms need a work-around for low values of $F1$.

This drawback was overcome by the finding that the global statistical properties of the waveform change with glottal opening and closing as well. We will come back to this issue in Sect. 10.7.1.

Structural analysis of the signal itself or of some simplified representation, especially when many possible structures have to be reviewed, is a good candidate for self-organizing, nonlinear pattern-recognition methods, i. e., for artificial neural networks. Such a PDA for speech was introduced by *Howard* and *Walliker* [10.63]. The speech signal is divided into nine subbands with a subsequent half-wave rectification and second-order linear smoothing in each channel. The underlying idea is to obtain a representation similar to that in a wide-band spectrogram. The basic extractor consists of a four-layer perceptron structure, the input layer consisting of 41 successive samples in each subband. Two hidden layers with 10 units each and a fully connected network are followed by a one-unit output layer, which is intended to yield an impulse when the network encounters a signal structure associated to the instant of glottal closure. The network is trained using (differentiated) output signals of a laryngograph as reference data. Such a structure has the advantage that it can be based upon several features occurring at different instants during a pitch period.

10.3.4 Cascaded Solutions

Among the many possibilities of such solutions, one is of particular interest here: the cascade of a robust short-term PDA and an accurate but less-robust time-domain PDA. Such an algorithm is further described in Sect. 10.7.1.

10.4 A Short Look into Voicing Determination

The task of voicing determination of speech signals may be split up into two subtasks: (1) a decision of whether or not a voiced excitation is present and (2) a decision of whether or not a voiceless excitation is present. If neither of these excitations is active, the current segment represents pause or silence; if both excitations are simultaneously active, we speak of mixed excitation. The two features *voiced* and *voiceless* are binary unless they occur simultaneously. In segments with mixed excitation the degree of voicing – for instance, the energy ratio of the voiced and voiceless excitations – may play a role, although this feature is rarely exploited.

Most voicing determination algorithms (VDAs) thus apply decisions. VDAs exploit almost any elementary speech signal parameter that may be computed independently of the type of input signal: energy, amplitude, short-term autocorrelation coefficients, zero-crossings count, ratio of signal amplitudes in different subbands or after different types of filtering, linear prediction error, or the salience of a pitch estimate. According to the method applied, VDAs can be grouped into three major categories: (1) simple threshold analysis algorithms, which exploit only a few basic parameters [10.64]; (2) more-complex algorithms based on pattern recognition methods; and (3) integrated algorithms for both voicing and pitch determination.

In this section, we distinguish between *voiceless* and *unvoiced*. Unvoiced means that a frame can be anything but voiced, i. e., it can be voiceless or silence. Voiceless means that voiceless excitation is present so that the frame is neither voiced nor silence. We will not review such algorithms that distinguish between speech and nonspeech – many of such algorithms have been developed for other applications, such as voice over Internet protocol (IP) or bandwidth reduction in mobile telephony (see, for instance, *Davis* et al. [10.65], for a survey). The basic task of the VDA in the context of pitch determination is to decide whether a frame or signal segment is voiced (and thus subject to pitch determination) or unvoiced.

10.4.1 Simultaneous Pitch and Voicing Determination

A number of PDAs – usually pertaining to the short-term analysis category – permit estimation the salience of their results without having to know the actual value of the pitch estimate. This is always possible when the amplitude of the significant maximum or minimum at T_0 or F_0 in the basic extractor of the PDA can be compared to a reference value. As an example, the ratio of the values of the autocorrelation function at $d = T_0$ and at $d = 0$ (the latter equalling the signal energy) gives a direct measure of the degree of periodicity of the signal. It is dangerous, however, to rely on this feature alone. Of course, it is correct (and almost trivial) to state that pitch can exist only when the signal is voiced. However, this statement cannot be simply reversed; i. e., we cannot say that a segment is unvoiced because pitch is not existent (or not measurable). The corresponding PDA may momentarily fail, or the signal may be voiced but irregular ([10.47, 66]; see also Fig. 10.2 or Sect. 10.1.2). Amplitude changes and especially small intraframe changes of F_0 severely degrade the *salience* of the pitch estimate [10.36]. It is thus at least necessary to check adjacent frames before making a decision [10.10]. In this respect, such a VDA behaves very much like a median smoother in pitch determination.

In principle, these VDAs do not make a voiced–unvoiced discrimination; rather they check for the presence of a (sufficient but not absolutely necessary) condition for a voiced signal. An improvement is to be expected when such criteria are only used for declaring a frame as voiced, and when the decision to declare it as unvoiced is based on additional criteria [10.67].

The VDA by *Lobanov* [10.68] avoids this problem, although it is based on a similar principle. A voiceless segment of speech represents a stochastic signal which is continuously excited. In contrast, the excitation of a voiced signal is confined to a few instants per period; major parts of the pitch period are composed of exponentially decaying oscillations, and adjacent samples of the signal are highly correlated. This contrast of a highly stochastic versus a highly deterministic signal is preserved even when a voiced signal becomes irregular or aperiodic. Lobanov's VDA exploits this feature by computing the Hilbert transform of the speech signal, combining the original signal and its Hilbert transform to yield the complex analytic signal, and plotting the momentary amplitude and phase of the analytic signal in the so-called phase plane. For voiced frames the analytic signal will describe a closed curve. During unvoiced segments, where the signal and its Hilbert transform are much less correlated, the curve will touch almost any point in the phase plane within a short interval. In Lobanov's algorithm the phase plane is crudely quantized, and the algorithm simply counts the number of points which have been touched within a given frame.

Talkin's PDA [10.17] integrates the VDA into the postprocessor that applies dynamic programming. Among the various estimates for T_0 to be tracked, there is always a candidate *unvoiced*, which is selected when it lies on the optimal path (Sect. 10.5.4).

Ahmadi and *Spanias* [10.67] present an improved VDA module within an implementation of the cepstrum PDA [10.10] for telephone-bandwidth speech. An utterance is processed in two passes. The first pass, covering the whole utterance, is to derive gross initial thresholds for a rough voiced–unvoiced decision. Distributions are taken for the relative amplitude of the main cepstral peak, the relative zero-crossings rate, and normalized signal energy. The medians of these distributions serve as initial thresholds for the decisions to be made in the second pass. A frame is roughly declared unvoiced if its energy and cepstral peak amplitudes are below and its zero crossings rate is above the respective threshold. Frames are declared voiced according to their cepstral peak amplitudes and a continuity criterion. The algorithm was evaluated on data from the TIMIT corpus; reference values were obtained using the PDA by *McAulay* and *Quatieri* [10.51] with visual inspection of uncertain frames. For clean speech, voiced-to-unvoiced and unvoiced-to-voiced errors together were about 1.5%.

McAuley and *Quatieri* [10.51] use their harmonic-model PDA (Sect. 10.2.5) to incorporate a VDA. It is based on the energy ratio between the harmonic energy and the energy of the nonharmonic part of the signal (the *noise*) which consists of everything not captured by the harmonic structure. Frames for which this ratio is above 10 dB are certainly voiced, while those for which the ratio is below 4 dB are certainly unvoiced.

Fisher et al. [10.69] start from a generalized log likelihood measure that is separately and independently evaluated for the two hypotheses that the frame is (1) voiced, or that it is (2) unvoiced. The measure for the frame being voiced is based on the aforementioned ratio between harmonic and nonharmonic energy, whereas the measure for unvoiced is based on a model of colored Gaussian noise. The hypothesis with the higher likelihood value wins for each frame; a dynamic-programming postprocessor (Sect. 10.5.4) integrates the VDA into the PDA which is also based on the harmonic-plus-noise model.

10.4.2 Pattern-Recognition VDAs

One of the motivations for applying a pattern-recognition algorithm in a VDA was the wish to get away from the conjunction of voicing determination and pitch determination [10.70]. The VDA by *Atal* and *Rabiner* [10.70] (the first of a series of VDAs developed at Bell Laboratories in the late 1970s) uses a statistical classifier and is based on five parameters: the signal energy, the zero-crossing rate, the autocorrelation coefficient at a delay of one sample, the first predictor coefficient of a 12 pole LP analysis, and the energy of the normalized prediction error. For a given environmental condition the algorithm works well, but it is rather sensitive to environmental changes, e.g., from high-quality speech to a telephone channel [10.47].

The usual classification problems in speech recognition, where we have to cope with a large number of different classes, require that the input parameters form specific clusters in the parameter space, which are then separated by the classifier. In contrast, the voicing determination problem has at most four categories (silence, voiced, voiceless, and mixed) and the distribution of the patterns in the parameter space is rather diffuse. It is thus appropriate to concentrate the VDA on patterns that are situated at or near the boundaries between the different categories in the parameter space. Such a VDA was developed by *Siegel* and *Bessey* [10.66]. For some applications, such as high-quality analysis–synthesis systems, incorporation of a mixed excitation source is desirable: (1) for voiced fricatives, and (2) for some high vowels, which tend to become partly devoiced in connected speech [10.71]. Siegel and Bessey further found that for the voiced–voiceless–mixed classification, the number of features used for a voiced-unvoiced classifier is insufficient. Their VDA is realized in two steps using a binary decision tree structure. The first step is a classifier which separates frames that are predominantly voiced from those that are predominantly unvoiced. It uses a minimum-distance statistical classifier exploiting seven features: normalized autocorrelation coefficient at unit sample delay, minimum normalized LP error, zero-crossings rate, signal energy, overall degree of periodicity (via AMDF), and degree of periodicity measured via the cepstrum in two subbands. In both categories the mixed frames are split off during the second step. The voiced–mixed decision uses another six features, mostly cepstral and LP measures, whereas the voiceless–mixed decision is based on two features alone. The VDA is reported to work with 94% overall accuracy and 77% correct identification of the mixed frames.

10.5 Evaluation and Postprocessing

To evaluate the performance of a measuring device, one should have another instrument with at least the same accuracy. If this is not available, at least objective criteria – or data – are required to check and adjust the behavior of the new device. In pitch and voicing determination both these bases of comparison are tedious to generate. There is no VDA or PDA that operates without errors [10.47]. There is no reference algorithm, even with instrumental support, that operates completely without manual inspection or control [10.8, 72]. Yet nowadays speech databases with reference pitch contours and voicing information have become widely available so that at least reliable reference data are there and are being used for evaluation.

In this section, we first deal with the question of how to generate reference data (Sect. 10.5.1). Then we consider the question of error analysis (Sect. 10.5.2) and present the results of some comparative evaluations (Sect. 10.5.3). Finally, we describe the problem of pitch tracking (Sect. 10.5.4), which is the foremost task of the postprocessor.

10.5.1 Developing Reference PDAs with Instrumental Help

A number of evaluations compared the algorithm(s) to be tested to the results of a well-known algorithm such as the cepstrum PDA, whose performance was known to be good. *Rabiner* et al. [10.47] used an interactive PDA to generate reference data. This procedure proved reliable and accurate but required a great deal of human work. Other evaluations [10.1] used the output signal of a vocoder for which the pitch contour was exactly known or the output signal of a mechanic accelerometer which derives the information on pitch from the vibrations of the neck tissue at the larynx. The latter device [10.73] was used by *Viswanathan* and *Russell* [10.74] for their evaluation of five PDAs. *Indefrey* et al. [10.34] used a laryngograph to obtain the signal for generating a reference contour.

Among the algorithms used for determining a reference pitch contour, methods that make use of an instrument (such as a mechanic accelerometer or a laryn-

gograph) that derives pitch directly from the laryngeal waveform have been shown to be most effective. This type of algorithm avoids most errors pertinent to the problem of pitch determination from the speech signal, and permits using natural speech for the evaluation of the performance of PDAs. Among the many instruments available, the laryngograph [10.72,75] is especially well suited for this kind of application. It is robust and reliable, does not prevent the speaker from natural articulation, and gives a good estimate for the instant of glottal closure. A number of PDAs have been designed for this device [10.8,72]. In addition, *Childers* et al. [10.76] propose a four-category VDA that exploits the speech signal and the laryngogram.

The principle of the laryngograph [10.75] is well known. A small high-frequency electric current is passed through the larynx by a pair of electrodes that are pressed against the neck at the position of the larynx from both sides. The opening and closing of the glottis during each pitch period cause the laryngeal conductance to vary; thus the high-frequency current is amplitude modulated. In the receiver the current is demodulated and amplified. Finally, the resulting signal is high-pass filtered to remove unwanted low-frequency components due to vertical movement of the larynx.

Figure 10.10 shows an example of the laryngogram (the output signal of the laryngograph) together with the corresponding speech signal. In contrast to the speech signal, the laryngogram is barely affected by the instantaneous configuration of the vocal tract, and the changes in shape or amplitude are comparatively small. Since every glottal cycle is represented by a single pulse, the use of the laryngograph reliably suppresses gross period-determination errors. In addition, it supplies the basis for a good voiced–unvoiced discrimination since the laryngogram is almost zero during unvoiced segments, when the glottis is always open. Nonetheless, the laryngograph is not free from problems: it may fail temporarily or permanently for some individual speakers, or it may miss the beginning or end of a voiced segment by a short interval – for instance, when the vocal folds, during the silent phase of a plosive, continue to oscillate without producing a signal, or when voicing is resumed after a plosive and the glottis does not completely close during the first periods [10.72]. For such reasons, visual inspection of the reference contour is necessary even with this configuration; these checks, however, can be confined to limited segments of the signal.

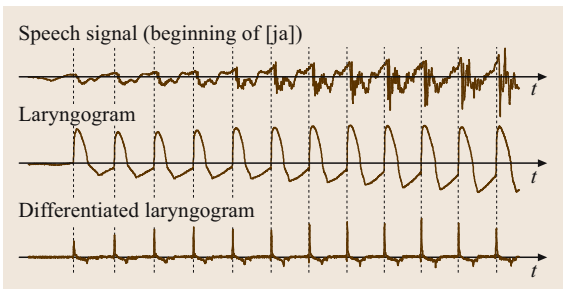


Fig. 10.10 Speech signal, laryngogram, differentiated laryngogram, and instants of glottal closure

The instant of glottal closure is the point of maximum vocal-tract excitation, and it is justifiable to define this instant as the beginning of a pitch period. In the laryngogram this feature is well documented. As long as the glottis is open, the conductance of the larynx is at a minimum and the laryngogram is low and almost flat. When the glottis closes, the laryngeal conductance goes up and the laryngogram shows a steep upward slope. The point of inflection during the steep rise of the laryngogram, i. e., the instant of maximum change of the laryngeal conductance, was found best suited to serve as the reference point for this event.

10.5.2 Error Analysis

According to the classic study by *Rabiner* et al. [10.47], which established the guidelines for the performance evaluation of these algorithms for speech, PDAs and VDAs commit four types of errors:

1. Gross pitch-determination errors
2. Fine pitch-determination errors, i. e., measurement inaccuracies
3. Voiced-to-unvoiced errors
4. Unvoiced-to-voiced errors

The latter two types represent errors of voicing determination, whereas the former two refer to pitch determination.

Gross pitch-determination errors are *drastic failures of a particular method or algorithm to determine pitch* [10.47]. Usually an error is considered to be gross when the deviation between the correct value of T_0 or F_0 and the estimate of the PDA exceeds the maximum rate of change a voice can produce without becoming irregular (*Rabiner* et al. [10.47]: 1 ms; *Hess* and *Indefrey* [10.8], *Mousset* et al. [10.77]: 10%; *Krubsack* and *Niederjohn* [10.78]: 0.25 octave). Typical gross errors are confusions between F_0 and the formant $F1$, which usually falls into the measuring range. Other typical errors are the so-called octave errors, i. e., taking $F_0/2$ or $2F_0$ as the pitch estimate.

On the other hand, error situations may also arise from *drastic failures of the voice to produce a regular excitation pattern*, which is not very frequent in well-behaved speech but is nearly always the case when the voice temporarily falls into creak or vocal fry. *Hedelin* and *Huber* [10.18] distinguish between four main types of irregularity of phonation that occur frequently in running speech: (1) laryngealization (a temporal near-closure of the glottis resulting in single, irregular glottal pulses); (2) creak or vocal fry as a temporal voice register (Fig. 10.2); (3) creaky voice, which is even less structured than creak; and (4) diplophonic excitation, which shows an irregular pattern between adjacent periods but a more-regular one between every second pitch period. A further problem, which may sometimes become severe, is the rate of change of fundamental frequency. *Xu* and *Sun* [10.79], also referring to earlier studies, give data for the maximum rate of change of F_0 that a human voice is able to produce without becoming irregular. They found that a human voice can change its F_0 at a speed up to 100 semitones per second, and that this limit is frequently reached during running speech. One hundred semitones per second means one semitone (6%) per 10 ms or two semitones (12%) per 20 ms. According to *Sreenivas* [10.36], a 4% within-frame F_0 change already affects the salience of the estimate in the same way as additive noise with 15 dB SNR. As we see from these data, a 10% change of F_0 within a frame can easily occur. If we interpret these data with respect to individual pitch periods, we see that deep male voices with long periods (10 ms and more) are more strongly affected than female voices. Nonetheless, a deviation of 10% for F_0 estimates between adjacent frames seems reasonable as a lower bound for gross errors because a larger change is beyond the capabilities of a human voice.

Hence, gross errors arise mainly from three standard situations.

- Adverse signal conditions: strong first formants, rapid change of the vocal-tract position, band-limited or noisy recordings. Good algorithms reduce these errors to a great extent but cannot completely avoid them [10.47].
- Insufficient algorithm performance, e.g., mismatch of F_0 and frame length [10.21]; temporary absence of the key feature in some algorithms.
- Errors that arise from irregular excitation of voiced signals. Since most algorithms perform some averaging or regularity check, they can do nothing but fail when the source becomes irregular.

When a PDA is equipped with an error-detecting routine (the majority of cases, even if no postprocessor is used), and when it detects that an individual estimate may be wrong, it is usually not able to decide reliably whether this situation is a true measurement error, which should be corrected or at least indicated, or a signal irregularity, where the estimate may be correct and should be preserved as it is. This inability of most PDAs to distinguish between the different sources of error situations is one of the great unsolved problems in pitch determination.

In the study by *Rabiner* et al. [10.47] gross errors are simply counted, and the percentage of frames with gross errors compared to the total number of (correctly recognized) voiced frames is given as the gross error rate. However, the perceptual importance of gross errors depends on the deviation between the estimate and the correct value as well as the energy of the frame [10.74, 80], from which a weighted gross error measure was derived [10.67],

$$\mathrm{GPE} = \frac{1}{K}\sum_{k=1}^{K}\left(\frac{E(k)}{E_{\max}}\right)^2 \left|\frac{p(k)-F_0(k)}{F_0(k)}\right| , \quad (10.21)$$

where $p(k)$ is the incorrect estimate, $E(k)$ is the energy of the frame, and $E_{\max}$ is the maximum energy in the utterance. It appears useful to include both these measures in an evaluation. The gross error count evaluates the PDA performance from a signal-processing point of view, whereas GPE says something about their perceptual relevance.

Measurement inaccuracies cause noisiness of the obtained T_0 or F_0 contour. They are small deviations from the correct value but can nevertheless be annoying to the listener. Again there are three main causes.

- Inaccurate determination of the key feature. This applies especially to algorithms that exploit the temporal structure of the signal, for instance, when the key feature is a principal maximum whose position within a pitch period depends on the formant $F1$.
- Intrinsic measurement inaccuracies, such as the ones introduced by sampling in digital systems.
- Errors from small fluctuations (*jitter*) of the voice, which contribute to the perception of *naturalness* and should thus be preserved.

Voicing errors are misclassifications of the VDA. We have to distinguish between voiced-to-unvoiced errors, in which a frame is classified as unvoiced although it is in fact voiced, and unvoiced-to-voiced errors, with the opposite misclassification. This scheme, as established

in [10.47], does not take into account mixed excitation. Voiced-to-unvoiced and unvoiced-to-voiced errors must be regarded separately because they are perceptually inequivalent [10.74], and the reasons for such errors in an actual implementation may be different and even contradictory. A number of VDAs even allow a tradeoff between these two errors by adjusting their parameters.

10.5.3 Evaluation of PDAs and VDAs– Some Results

Due to the absence of reliable criteria and systematic guidelines, few publications on early PDAs included a quantitative evaluation. As this situation has thoroughly changed, publications presenting new PDAs increasingly also evaluate them. Mostly the newly developed PDA is evaluated against some well-known PDA(s) to show that the new approach is in some way or for some kind of signals and conditions better or at least equivalent to the known algorithms [10.46,52]. The *Keele* database [10.81] has played a major role in this respect. We will not discuss these evaluations here due to lack of space; we rather deal with a couple of studies that did not aim at developing a new PDA but were done to establish guidelines and show the state of the art.

The classic studies by *Rabiner* et al. [10.47] and *McGonegal* et al. [10.56] investigated seven PDAs (two time domain, five short-term analysis) with respect to pitch and voicing determination. The main results were:

- None of the PDAs investigated were error free, even under good recording conditions. Each PDA had its own favorite error; nevertheless, all error conditions occurred for all of the PDAs.
- Almost any gross error was perceptible; in addition, unnatural noisiness of a pitch contour was well perceived.
- The subjective evaluation did not match the preference of the objective evaluation. In fact, none of the objective criteria (number of gross errors, noisiness of the pitch contour, or voicing errors) correlated well with the subjective scale of preference.

Hence the question of which errors in pitch and voicing determination are really annoying for the human ear remained open. This issue was further pursued by *Viswanathan* and *Russell* [10.74], who developed objective evaluation methods that are more closely correlated to the subjective judgments. The individual error categories are weighted according to the consistency of the error, i. e., the number of consecutive erroneous frames, the momentary signal energy, the magnitude of the error, and the special context.

Indefrey et al. [10.34], concentrating on the evaluation of pitch determination errors only, investigated several short-term PDAs in various configurations. They showed that in many situations different short-term analysis PDAs behave in a complementary way so that combining them in a multichannel PDA could lead to better overall performance.

Indefrey et al. [10.34] also investigated the performance of double-transform PDAs (cf. Sect. 10.2.2) with additive Gaussian noise. Under this condition these algorithms tend to break down at SNRs between 0 and −6 dB. It does not make a big difference whether the SNR is defined globally (i. e., with a constant noise level over a whole utterance) or segmentally (i. e., with the same SNR for each frame), except that the slope of the error curve at the breakdown point is larger for segmental SNR. These results were confirmed in a number of other studies [10.26, 46, 52]. *Moreno* and *Fonollosa* [10.52] evaluated several autocorrelation PDAs (among them their own) with several kinds of noise signals and found that for the low-frequency-biased car noise the breakdown starts at an SNR of about 6 dB. The same holds for babble noise [10.46].

De Cheveigné and *Kawahara* [10.27] investigated eight PDAs whose software was available via the Internet together with two of their own developments ([10.27], cf. Sect. 10.2.1; [10.82], based on an IF principle). Only gross errors were considered. The evaluation was based upon an extensive database (almost two hours of speech) with samples from three languages (English, French, and Japanese), including falsetto speech from male speakers, and laryngograms as reference signals. Obviously aperiodic voiced signals were excluded. Postprocessors and VDAs were disabled in the algorithms as far as possible. The evaluation showed great differences between algorithms and partly rather bad performance (more than 10% gross errors for some of them); the best one produced about 0.6% on average. The evaluation also showed considerable dependency of the error rate on the data so that the authors claim the need for large databases when performing such evaluations.

All these evaluations show that there is still no PDA that works without errors, although they work better now than 20 years ago. A gross error count of 0.6% is regarded as excellent; nonetheless we must not forget that, with the usual frame rate of 100 frames per second, such an algorithm still produces a grossly wrong estimate every two seconds of speech on average.

10.5.4 Postprocessing and Pitch Tracking

A standard procedure for the reduction of pitch determination errors is smoothing. Smoothing is possible when a pitch contour is given as a sequence of T_0 or F_0 estimates and not as delimiters of individual periods. The two standard smoothing methods are linear smoothing using some kind of low-pass filter and (nonlinear) median smoothing [10.83]. Linear smoothing reduces measurement inaccuracies but is unable to cope with the effect of gross pitch determination errors, which are reduced in size and distributed over a larger amount of time but are not really removed. Median smoothing, on the other hand, replaces each pitch estimate with the middle value of an ordered sequence of three, five, or seven adjacent estimates; gross outliers are removed, but measurement inaccuracies remain unchanged. *Rabiner* et al. [10.83] combine these methods and propose a two-step smoothing procedure with median smoothing coming first, followed by a linear smoother. Linear smoothing, however, can be dangerous since it may replace a gross error that has been left in the median-smoothed contour by some estimates lying between the correct value and the error and so cause an inflection in the contour that is not due to the signal.

Applying such a smoothing algorithm was shown to substantially improve the (objective and subjective) performance of any PDA to which it was applied [10.47, 56]. *Specker* [10.84] showed that postprocessing is able to reduce the number of gross errors in a time-domain PDA by almost an order of magnitude.

Secrest and *Doddington* [10.80] used dynamic programming methods to find an optimal path through a list of pitch estimate candidates with the smoothness of the contour as the major criterion. They showed that this technique performed better than any linear, nonlinear, or median smoothing. This approach was further developed by *Talkin* [10.17]. Dynamic programming is well suited to pitch tracking since it allows the basic extractor to give several pitch candidates so that we can deal with more than only the best choice in each frame. Each candidate is accompanied by a salience measure (usually the relative amplitude of the corresponding peak in the representation from where the estimate is derived, with respect to the reference point, e.g., the value of the ACF at zero lag). In addition, Talkin's PDA supplies one candidate *unvoiced* per frame. Pitch tracking is done by searching for an optimal path through the candidates from consecutive frames by minimizing a global cost function. This global cost function is formed as the sum of weighted local per-frame cost functions of two types: (1) candidate costs, and (2) transition costs between consecutive frames.

Candidate costs distinguish between pitch candidates and the unvoiced candidate. The cost of a pitch candidate equals one minus the salience measure of this candidate. The cost of the unvoiced candidate is a constant penalty plus the maximum salience measure within the current frame.

The transition cost between consecutive frames also depends on voicing. Between two unvoiced candidates it is zero. Between two pitch candidates it depends on the difference in frequency between the two estimates, and special attention is given to octave jumps, which are made costly but not totally impossible. The costs for voiced-to-unvoiced transitions and vice versa include a term with the reciprocal *Itakura* distance [10.85], an energy term, and an extra penalty for this transition. The rationale is that (1) these transitions are not too frequent, (2) there are usually large spectral changes between a voiced and an unvoiced frame, and (3) energy usually decreases at the end of a voiced part and increases at its beginning.

There is no latency limit for the algorithm to find the optimal path; in principle the search can extend over a whole utterance. *Talkin* [10.17], however, reports that it rarely takes more than 100 ms for all possible paths to converge to a single point. The algorithm is part of the well-known ESPS software package for speech analysis. A comparable postprocessor operating on a probabilistic approach is described in [10.86].

10.6 Applications in Speech and Music

Applications for PDAs in speech can be grouped into four areas [10.1]:

1. Speech technology
2. Basic linguistic research, e.g., investigation of intonation
3. Education, such as teaching intonation to foreign-language learners or to hard-of-hearing persons

4. Clinical applications, such as the investigation of voice quality

Some of these application areas have changed greatly over the last two decades. The vocoder, which was the main application for earlier speech technology, has almost disappeared. Instead, investigating prosodic events such as intonation, particularly in spontaneous speech, has become an important issue in speech understanding systems [10.87], and many of these systems now contain a prosody module. As it is a long-term goal in speech technology to make such devices operable from almost anywhere, a PDA may even have to cope with signals from mobile phones, which can be extremely bad and inconsistent. Another new application area is data-driven speech synthesis. Algorithms for time-domain pitch modification, such as the well-known *pitch-synchronous overlap add* (PSOLA) algorithm [10.57], require precise pitch period determination to work properly, and with the recent technology of nonuniform unit selection synthesis large speech corpora have to be analyzed, yet usually with excellent-quality signals free from phase distortions.

What are the implications of this application shift for the development of algorithms? PDAs for precise pitch period determination of good-quality speech signals have been known for a long time; nonetheless the main problem is exact synchronization with laryngeal cycles, such as the instant of glottal closure. Such algorithms, which originally come from clinical applications where they were applied to isolated vowels, have been extended to work for running speech as well.

In prosody recognition, intonation research and speech technology now go together to a certain extent. Prosody recognition needs intonation contours, not individual periods, and a certain lag between the running time of the signal and the time of release of an estimate is tolerable, in contrast to a vocoder where the result must be available without delay. On the other hand, prosody recognition must rely on automatic estimates and cope with adverse conditions; above all, this requires robustness.

The number of devices available for computer-aided intonation teaching has been small [10.88]. However, with the increased use of high-quality PDAs for intonation research, this will change. In the clinical area, digital hearing prostheses have created a new application area [10.63, 89]. We cannot discuss these applications here for reasons of space; the reader is referred to [10.88, 89] for surveys.

Pitch determination of musical signals has three main application areas, two of which are closely related:

1. Automatic notation of melodies and tunes, and automatic scoring
2. Information retrieval from audio data
3. Real-time capturing of tone frequencies of musical instruments for musical instrument digital interface (MIDI) applications

Automatic notation of melodies is a long-standing problem. As early as 1979 *Askenfelt* [10.90] reported on a project to automatically transcribe folk melodies that had been collected in a large corpus. In this case most of the melodies were one-voiced so that PDAs developed for speech signals could be used.

This is not always the case for music, however, and we must therefore count as a particular problem in music that more than one note can be played at the same time, and that PDAs for this application may have to cope with multiple pitches and have to determine them all, if possible.

Information retrieval from audio data (audio data mining) also involves pitch determination of musical signals. This application area is closely related to the preceding one and also involves the problem of multiple pitches. An evolving application is the so-called query by humming. Consider a person who listens to an unknown tune in the radio, likes it, calls a call center with a musical database, hums or sings the melody, and wants the title of the song retrieved. This is a difficult task with pitch determination being a part thereof.

Transcription of a musical melody into musical notes is much more than mere pitch determination. One has to recover the rhythm from the timing of the melody, and one has to take care of the timing, i. e., when one note ends and another one starts. These questions, which are partly still open, go beyond the scope of pitch determination and will not be further discussed here.

Real-time capturing for MIDI is closely related to the old vocoder application in speech. A PDA is always desirable if a MIDI synthesizer is to be driven from an instrument other than a keyboard, e.g., from an electric guitar. Here the main problem is that the response must be almost instantaneous; a delay of even 50 ms could be detrimental for a live performance.

10.7 Some New Challenges and Developments

This section will concentrate on three problems:

1. Detecting the instant of glottal closure for applications in speech synthesis
2. Multiple pitch detection, particularly in music [10.93]
3. Instantaneousness versus reliability

10.7.1 Detecting the Instant of Glottal Closure

If a PDA in speech is required to be very accurate, it is optimal to synchronize it with the instants of glottal closure. A cascaded PDA for this purpose was developed by *Hess* [10.92] based on a previous algorithm by *Cheng* and *O'Shaughnessy* [10.91]. It is intended for work in time-domain speech synthesis and determines the glottal closure instant (GCI) from undistorted signals. The first part of the cascade is a short-term PDA applying a double-spectral-transform principle [10.34]. The second part uses the estimate of the first PDA to restrict its momentary F_0 range to an octave around this value.

According to *Mousset* et al. [10.77] a GCI detector consists of four steps:

1. Acoustic speech signal pre-emphasis (optional)
2. A transformation aiming to produce peaks at GCIs
3. Postprocessing aiming to increase contrast in the resulting signal (optional)
4. A peak picking operation

The algorithm described here follows this scheme.

In the source-filter approach the speech signal is the response of the supraglottal system to the pulse train generated by the source, and a pitch period can be regarded as the beginning of the impulse response of the vocal tract. If we now compute the correlation function $c(n)$ between the speech signal $s(n)$ and the beginning of the pertinent impulse response $h(n)$ of the vocal tract,

$$c(n) = \sum_k s(n+k)h(k) \qquad (10.22)$$

there will be maxima at those instants n that correspond to an GCI. The impulse response $h(n)$ is estimated via linear prediction. The main peaks of $c(n)$ synchronize well with the individual GCIs. Strong formants, however, also show up in $c(n)$, and further processing is required. Cheng and O'Shaughnessy suggested that one should calculate the envelope of $c(n)$, send it through a high-pass filter and a half-way rectifier to enhance the leading amplitudes at the beginning of each pitch period, and multiply $c(n)$ by the resulting waveform $d(n)$. The envelope

$$e(n) = \sqrt{[c(n)]^2 + [c_H(n)]^2}\,, \qquad (10.23)$$

where $c_H(n)$ is the Hilbert transform of $c(n)$, is calculated using a digital Hilbert filter. Figure 10.11 shows an example.

This approach has the problem known from all PDAs that apply some sort of inverse filtering (cf. Sect. 10.3.3) that it fails systematically when the formant $F1$ coincides with F_0 and the signal becomes almost sinusoidal. Then $e(n)$ becomes a constant, and $d(n)$ is either zero or fluctuates at random around zero instead of displaying the envelope of a damped formant oscillation. A way out of this problem is to partly bridge the high-pass filter so that the constant component of the envelope is not completely removed. If the short-term analysis enters a reliable estimate of T_0, rather stringent correction routines can remove the unwanted peaks in $c(n)$ and at the same time preserve correct processing. The exact position of the GCIs is derived from $c(n)$ anyway.

Group delay methods have also been proposed for GCI determination [10.94, 95]. Group delay is defined as the derivative of the phase spectrum with respect to

Fig. 10.11a–e Determining the instant of glottal closure via a maximum-likelihood criterion [10.91, 92]: example of performance. **(a)** Signal frame (45 ms); **(b)** impulse response $h(n)$ of the vocal tract, estimated by linear prediction; **(c)** correlation function $c(n)$; **(d)** envelope $e(n)$ (dotted line) and high-pass filtered envelope; **(e)** product of $c(n)$ and the filtered envelope

frequency and can be calculated [10.94] via the DFT:

$$\tau_{\mathrm{G}}(m) = \mathrm{Re}\left(\frac{\tilde{S}(m)}{S(m)}\right) ; \qquad (10.24)$$

$$S(m) = \mathrm{DFT}\{s(n)\} ; \quad \tilde{S}(m) = \mathrm{DFT}\{ns(n)\}$$

If a frame contains a single impulse at position $k = n_0$, this will produce a constant group delay of n_0 for all frequency indexes m. If there is additional noise in the frame, one can expect that at least the average group delay (averaged over all frequencies) equals n_0. If we now compute the average group delay for each sample of the time signal $s(n)$, it will show a negative-going zero crossing (with a slope of -1) when the impulse is at the starting position of the frame. GCIs mark nonstationarities in the signal and show up as impulse-like structures, for instance, in an LP residual. This makes the method useful for GCI detection, and there are algorithmic shortcuts that allow the average group delay to be computed without needing two DFTs per sample. *Brookes* et al. [10.95] investigate several weighted measures for the average group delay and find that these measures are quite robust against additive noise; however, they critically depend on the length of the time-domain window used for group delay measurement. If there is no impulse in the frame, the average group delay will vary around zero; if there is more than one impulse, the results are unpredictable. In a cascaded implementation using an auxiliary short-term PDA, when an estimate for T_0 is available, it is straightforward to adjust the window length according to this estimate.

Another method, rather simple to implement, is based on singular value decomposition. *Ma* et al. [10.96] show that the Frobenius norm $||\mathbf{S}||_{\mathrm{F}}$ of an $N \times (K+1)$ matrix $\mathbf{S}$, being

$$||\mathbf{S}||_{\mathrm{F}} = \sqrt{\sum_{n=1}^{N}\sum_{k=1}^{K+1} s_{nk}^2} \quad \text{with} \qquad (10.25)$$

$$\mathbf{S} = (s_{nk};\ 1 \le n \le N;\ 1 \le k \le K+1)$$

equals the product of the squared singular values of the singular value decomposition (SVD) of $\mathbf{S}$,

$$||\mathbf{S}||_{\mathrm{F}} = \sqrt{\sum_{k=1}^{K+1} \sigma_k^2} \, . \qquad (10.26)$$

SVD is known to model linear dependencies of the rows and columns of the associated matrices. GCIs provide new information and cannot be covered by linear modeling. Hence, singular values tend to be large when the pertinent speech data matrix representing a signal frame,

$$\mathbf{S} = \begin{pmatrix} s(K) & s(K-1) & \cdots & s(0) \\ s(K+1) & s(K) & \cdots & s(1) \\ \vdots & \vdots & \ddots & \vdots \\ s(K+N-1) & s(K+N-2) & \cdots & s(N-1) \end{pmatrix}$$

contains a glottal impulse. (We assume $N > K$ and full column rank of the matrix $\mathbf{S}$ [10.96].) The singular values become largest when the glottal impulse is found in the first row of $\mathbf{S}$. So the Frobenius norm gives an algorithmic shortcut to the costly SVD of $\mathbf{S}$ which would have to be performed otherwise. This algorithm had a couple of forerunners that are well described in [10.96].

Other methods to determine the GCI involve neural networks with appropriate training [10.63], wavelet functions [10.97], simplified inverse filtering [10.77], nonlinear filtering [10.98] or statistical evaluation of the nonstationarity of the signal at the GCI [10.99]. *Mousset* et al. [10.77] evaluated several of these methods using the Keele database for PDA evaluation [10.81]. They used the analysis scheme proposed by *Rabiner* et al. [10.47] and extended it by two error classes suitable for any time-domain PDA: (1) insertion of a GCI marker where no GCI is present, and (2) miss (deletion) of a GCI that should have been detected. Their results show that the methods evaluated are about equivalent but show some sensitivity to the length of the respective time windows and to the speakers.

10.7.2 Multiple Pitch Determination

Simultaneous tones with different frequencies require some frequency-domain processing since, in this domain, peaks resulting from different tones show up at their respective frequencies. The same holds for the lag domain of a double-spectral-transform PDA when the nonlinear distortion in the frequency is of local nature, such as in computing a cepstrum or an autocorrelation function. Hence PDAs that explicitly or implicitly involve a Fourier transform will have this property, as was shown in [10.33] using simultaneous speech from two talkers as well as musical signals. Although not explicitly stated in the literature, active modeling would also allow multipitch tracking.

Such PDAs have been used in several configurations for speaker separation. The usual procedure is that the PDA determines pitch for one of the speakers; then a speech enhancement procedure (e.g., spectral subtraction) is applied to remove this speaker from the signal. The PDA then determines the pitch of the sec-

ond speaker. The problem with this configuration is the correct assignment of the signal to the two speakers, particular when the signal from a speaker is unvoiced. *De Cheveigné* [10.100] describes a PDA designed for this purpose that estimates two pitches simultaneously using a cascaded comb filter and a two-dimensional AMDF estimation procedure.

Multiple pitch determination, particularly for music, is a wide field that would justify a chapter of its own. In this section we can give only a small selection of examples. For more-comprehensive surveys of the activities in this field, the reader is referred to *De Cheveigné* [10.100] or *Klapuri* [10.101].

Goto's algorithm [10.102] for music signals focuses on extracting *leading voices* from a polyphonic signal, in this case a melody line and a bass line which occupy different and non-overlapping F_0 ranges (32–260 Hz versus 260–4100 Hz). The PDA works in the frequency domain. It basically estimates the fundamental frequency of the most predominant harmonic structure corresponding to the melody or bass line. It simultaneously takes into account all possibilities for F_0 and treats the input spectrum as if it contains all possible harmonic structures with different weights (amplitudes). It regards a probability density function (PDF) of the input frequency components as a weighted mixture of harmonic-structure tone models (represented by PDFs) of all possible pitches and simultaneously estimates both their weights corresponding to the relative dominance of every possible harmonic structure and the shape of the tone models by maximum a-posteriori probability (MAP) estimation regarding their prior distribution. It then considers the maximum-weight model as the most predominant harmonic structure and obtains its fundamental frequency. A multiple-agent architecture evaluates the temporal continuity of the estimates.

The PDA uses a logarithmic frequency scale corresponding to the musical notation unit *cent*,

$$\frac{f}{\text{cent}} = 1200 \log_2 \frac{f/\text{Hz}}{440 \cdot 2^{\frac{3}{12}-5}} , \qquad (10.27)$$

where 1 cent equals 1/100 of a tempered semitone so that an octave consists of 1200 cents. Each tone model corresponds to a trial fundamental frequency p and provides a harmonic structure; the individual harmonics are modeled as weighted one-dimensional Gaussian distribution. The weights of all models are estimated simultaneously, where the same frequency can be shared by different harmonics of different tone models. The tone model with maximum weight yields the estimate for the predominant F_0. A multiple-agent architecture performs a temporal track across frames and gives the most stable trajectory as the final result. It consists of a salience detector that picks promising F_0 candidates, and a number of agents that interact to allocate the salient peaks among themselves according to peak closeness. Each agent has its own penalty record, and it gets a penalty when no suitable peak can momentarily be allocated to it. If a penalty threshold is exceeded, the agent is terminated. If a peak cannot be assigned to a running agent, a new agent is created. The final output is determined on the basis of which agent has the highest reliability and greatest total power along the trajectory of the peak it is tracking. Detection rates for melody and bass are reported as 80–90% and depend on the respective tune being processed.

The PDA by *Tolonen* and *Karjalainen* [10.103] uses a perception-oriented double-transform PDA based on the unitary model of pitch perception [10.6]. The original model in [10.6] employs a large number of critical-band filters (spaced at much less than a critical band). In each channel the signal is half-wave rectified and low-pass filtered, and its short-term ACF is determined. The ACFs of all channels are then added to give an estimate of pitch. In Tolonen and Karjalainen's PDA the signal is first filtered by an optional pre-whitening filter and then split into only two subbands with a cutoff frequency of 1000 Hz. The high-frequency channel is then rectified and smoothed. The lag-domain representations obtained by the double-spectral-transform principle for each subband are added up to the so-called summary ACF. This function is then enhanced. It is first clipped from under so that only positive values are retained. Then it is time stretched by a factor of 2, 3, etc. and subtracted from the original summary ACF, and again only positive values are retained. From this enhanced summary ACF significant maxima are sought. For musical sound analysis, relatively long windows (up to 180 ms) are employed. The algorithm works well when the simultaneous tones in the input signal do not differ too much in amplitude. The method is limited to fundamental frequencies below 1000 Hz [10.101].

Klapuri [10.101], besides giving a comprehensive survey of PDAs for multipitch determination, developed two PDAs for this task. One of these is based on the unitary model of pitch perception [10.6], with some modification that makes the PDA more reliable and computationally less costly. The other is based on an iterative frequency-domain technique. The predominant pitch is detected, then the corresponding harmonic structure is removed from the spectrum, and the procedure is repeated for the next predominant pitch. The

iteration is stopped when the energy of the harmonic structure drops below a given threshold.

Determination of the predominant harmonic structure follows a perception-oriented approach that resembles Terhardt's virtual pitch model [10.4] in that it is tolerant toward inharmonicity. The spectrum is subdivided into 18 subbands with overlapping triangular transfer functions which add to unity for adjacent subbands. In each subband salience estimates are made for each trial fundamental frequency p in the measuring range,

$$L(p,k) = \max_m \left(c(m,i) \sum_i^{(\text{in subband } k)} S(m+ip) \right) , \tag{10.28}$$

where m stands for a possible (small) offset due to inharmonicity, and c specifies a weighting function that depends on m and the harmonic number i. The salience estimates are then added over all subbands with the additional possibility of taking into account shift of higher harmonics toward higher frequencies, as it sometimes occurs with musical instruments. One of the trial frequencies p emerges as predominant, and the pertinent harmonic amplitude spectrum is smoothed. To remove the tone, the smoothed harmonic structure is subtracted from the overall spectrum.

Special attention is given to the problem of several tones with harmonic frequency ratios. The smoothing procedure helps to solve this problem. Think of two tones with a frequency ratio of 1:3. All the harmonics of the higher tone coincide with harmonics of the lower one, but we can expect that every third harmonic has an outstanding amplitude. It is likely that the lower tone will be detected first. The smoothing procedure equalizes these amplitude fluctuations so that the higher tone will be preserved when the lower one is removed.

The PDA by *Kameoka* et al. [10.104] performs simultaneous multipitch extraction in the frequency domain based on a statistical model given by

$$\{\Theta\} = \{\mu(k), w(k), \sigma | k = 1, \ldots, K\} . \tag{10.29}$$

The model consists of a set of K harmonic structures. Each of these is described by a tied Gaussian mixture (which corresponds to *Goto*'s PDA [10.102], see above). Its vector $\boldsymbol{\mu}$ of mean values stands for the frequencies of the partials and has only one degree of freedom, i. e., its fundamental frequency. The weight $w(k)$ stands for the predominance of the k-th harmonic structure; the variance is kept constant for the sake of simplicity. The model is initialized, and the parameters are iteratively improved using a log likelihood criterion and the expectation maximization algorithm [10.105]. As the true number of simultaneous tones in the signal is unknown, the model order K is initialized too high and becomes part of the optimization. Akaike's information criterion (AIC), which specifies a tradeoff between the order of a model and its accuracy,

$$\begin{aligned} \text{AIC} = {} & 2 \cdot (\text{number of free parameters}) \\ & - 2 \cdot (\text{maximum of log likelihood}) \end{aligned} \tag{10.30}$$

is used to determine the optimal value of K. A harmonic structure is abandoned when its weight becomes low or when it is placed between two other structures that move toward each other and have higher weights. If a harmonic structure is abandoned, the number of free parameters decreases by two (frequency and weight), and the likelihood gets worse. Whenever a harmonic structure is to be abandoned, AIC is computed. The iteration is stopped when AIC has reached a minimum.

The role of gross pitch determination errors in this context differs from that in speech. In speech we have the special problem of octave errors (cf. Sect. 10.5.2) which are to be strictly avoided. In polyphonic music, on the other hand, two instruments frequently play an octave apart from each other, and should then both be detected. *Klapuri*'s smoothing procedure [10.101] explicitly takes into account two tones at any harmonic interval. Nevertheless it is reported [10.101, 104] that the PDAs have problems when notes are played at certain musical intervals, among them the octave.

10.7.3 Instantaneousness Versus Reliability

It is always desirable to get the estimate from a PDA instantaneously. Processing time depends on two factors: (1) computational complexity of the PDA and speed of the device running it, and (2) latency, i. e., the amount of signal required to get a reliable estimate plus computing time. A PDA runs in real time if the processing time required is less than the elapsed time, say, between two successive frames. As today's computers are able to run even complex PDAs in real time, we can put aside the first issue. With the amount of signal required, however, there may be a hidden problem. An ordinary short-term PDA needs (at least) two complete pitch periods to detect periodicity in an orderly manner. If the spectral transform is done in one step for the whole range of F_0, latency will be twice the longest possible period in the window. For speech, with F_0 ranging from 50 Hz for a deep male voice up to, say, 1000 Hz for a child in spontaneous speech, this means a lag of at least 40 ms from

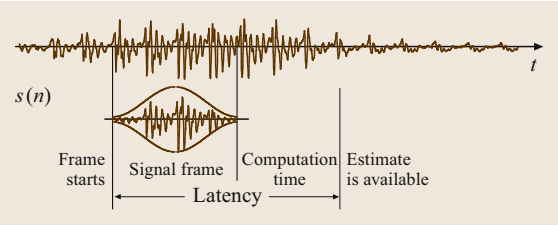


Fig. 10.12 The latency problem. Latency is defined as the elapsed time from the beginning of the frame until the estimate is available

the beginning of the window until we obtain an estimate, not including processing time. In vocoder telephony this may just be tolerable, but this will be a problem for online capturing of music in a MIDI application. Of course the latency will go down when the lower end of the frequency range gets higher. One period of the lowest tone of an ordinary guitar lasts about 12 ms, while that of a violin lasts about 5 ms. However, if the PDA applies postprocessing, for instance, pitch contour tracking by dynamic programming methods [10.17], latencies will go up drastically. It goes without saying that such methods cannot be applied when tough time constraints are given.

How can this problem be solved? If reliability requires a short-term analysis PDA, we must speed it up. If we can assume that the signal is nearly sinusoidal, we can apply time-domain methods, which may be more instantaneous.

Several attempts have been made to speed up short-term analysis PDAs. It has been known for a long time that distance functions, such as the AMDF, can work with short frames. Since no Fourier transform is involved, the function can be computed synchronously as time runs, and so we may be able to obtain estimates of T_0 in little more than the duration of the period to be measured. Estimates of short periods will thus be available sooner than those of long ones. Such nonstationary approaches do not only exist for distance functions. For instance, *Talkin*'s PDA ([10.17]; Sect. 10.2.1) uses a nonstationary autocorrelation function. *Yoo* and *Fujinaga* [10.106] performed some practical experiments with several hardware and software PDAs for MIDI capture on a violin and found latencies of 15–90 ms. For a MIDI application this may already be unacceptable.

If the signal is near-sinusoidal, time-domain pitch determination can be very fast. The lag between two consecutive zero crossings or between a maximum and the consecutive minimum of a sinusoid equals half a period, and only a quarter of a period elapses between an extreme (maximum or minimum) and the adjacent zero crossing. Even faster methods needing fewer signal samples can be conceived, leading to a kind of *anytime* algorithm that can yield a rather imprecise estimate almost instantaneously but is able to refine this estimate when given more time. (Remember, however, that in MIDI applications a tone command can be given only once; if it is in error, the wrong tone will come out.) In this case the problem is: how can we obtain a signal close to a sinusoid? In speech this is unrealistic due to the transfer function of the vocal tract, where lip radiation introduces a zero at $f = 0$ that strongly attenuates the first harmonic. In music, however, such an approach can be of interest when an instrument yields a signal that is almost sinusoidal, or when the capturing microphone or sensor is placed on the instrument in such a way that it performs as a low-pass filter.

10.8 Concluding Remarks

The problem of pitch determination and fundamental frequency tracking is long-standing, known, and yet unsolved in a general sense. However, for most applications, algorithms that yield good and acceptable solutions have been developed. Applications in speech and music have moved away from the vocoder toward prosody recognition, automatic melody detection, acoustic data retrieval, computational auditory scene analysis, and high-precision analysis of speech synthesis corpora. New challenges for the development of PDAs, among others, include high-precision pitch period determination, processing of signals with multiple pitches, and PDAs with very short latencies.

References

10.1 W.J. Hess: *Pitch Determination of Speech Signals – Algorithms and Devices* (Springer, Berlin, Heidelberg 1983)

10.2 R.J. McAulay, T.F. Quatieri: Speech analysis/synthesis based on a sinusoidal representation, IEEE Trans. Acoust. Speech Signal Process. **34**, 744–754 (1986)

10.3 E. Zwicker, W.J. Hess, E. Terhardt: Erkennung gesprochener Zahlworte mit Funktionsmodell und Rechenanlage, Kybernetik **3**, 267–272 (1967), (in German)

10.4 E. Terhardt: Calculating virtual pitch, Hearing Res. **1**, 155–182 (1979)

10.5 R. Plomp: *Aspects of Tone Sensation* (Academic, London 1976)

10.6 R. Meddis, L. O'Mard: A unitary model for pitch perception, J. Acoust. Soc. Am. **102**, 1811–1820 (1997)

10.7 K.J. Kohler: 25 Years of Phonetica: Preface to the special issue on pitch analysis, Phonetica **39**, 185–187 (1992)

10.8 W.J. Hess, H. Indefrey: Accurate time-domain pitch determination of speech signals by means of a laryngograph, Speech Commun. **6**, 55–68 (1987)

10.9 W.J. Hess: Pitch and voicing determination. In: *Advances in Speech Signal Processing*, ed. by M.M. Sondhi, S. Furui (Dekker, New York 1992), p.3–48

10.10 A.M. Noll: Cepstrum pitch determination, J. Acoust. Soc. Am. **41**, 293–309 (1967)

10.11 L.R. Rabiner: On the use of autocorrelation analysis for pitch detection, IEEE Trans. Acoust. Speech Signal Process. **25**, 24–33 (1977)

10.12 E. Terhardt, G. Stoll, M. Seewann: Algorithm for extraction of pitch and pitch salience from complex tonal signals, J. Acoust. Soc. Am. **71**, 679–688 (1982)

10.13 M.S. Harris, N. Umeda: Difference limens for fundamental frequency contours in sentences, J. Acoust. Soc. Am. **81**, 1139–1145 (1987)

10.14 J. 't Hart: Differential sensitivity to pitch distance, particularly in speech, J. Acoust. Soc. Am. **69**, 811–822 (1981)

10.15 H. Duifhuis, L.F. Willems, R.J. Sluyter: Measurement of pitch in speech: an implementation of Goldstein's theory of pitch perception, J. Acoust. Soc. Am. **71**, 1568–1580 (1982)

10.16 D.J. Hermes: Measurement of pitch by subharmonic summation, J. Acoust. Soc. Am. **83**, 257–264 (1988)

10.17 D. Talkin: A robust algorithm for pitch tracking (RAPT). In: *Speech Coding and Synthesis*, ed. by B. Kleijn, K. Paliwal (Elsevier, Amsterdam 1995), p.–495–518

10.18 P. Hedelin, D. Huber: Pitch period determination of aperiodic speech signals, Proc. IEEE ICASSP (1990) pp. 361–364

10.19 H. Hollien: On vocal registers, J. Phonet. **2**, 225–243 (1974)

10.20 N.P. McKinney: *Laryngeal Frequency Analysis for Linguistic Research* (Univ. Michigan, Ann Arbor 1965), Res. Rept. No. 14

10.21 H. Fujisaki, K. Hirose, K. Shimizu: A new system for reliable pitch extraction of speech, Proc. IEEE ICASSP (1986), paper 34.16

10.22 M.M. Sondhi: New methods of pitch extraction, IEEE Trans. Acoust. Speech Signal Process. **26**, 262–266 (1968)

10.23 J.D. Markel: The SIFT algorithm for fundamental frequency estimation, IEEE Trans. Acoust. Speech Signal Process. **20**, 149–153 (1972)

10.24 V.N. Sobolev, S.P. Baronin: Investigation of the shift method for pitch determination, Elektrosvyaz **12**, 30–36 (1968), in Russian

10.25 J.A. Moorer: The optimum comb method of pitch period analysis of continuous digitized speech, IEEE Trans. Acoust. Speech Signal Process. **22**, 330–338 (1974)

10.26 T. Shimamura, H. Kobayashi: Weighted autocorrelation for pitch extraction of noisy speech, IEEE Trans. Speech Audio Process. **9**, 727–730 (2001)

10.27 A. de Cheveigné, H. Kawahara: YIN, a fundamental frequency estimator for speech and music, J. Acoust. Soc. Am. **111**, 1917–1930 (2002)

10.28 K. Hirose, H. Fujisaki, S. Seto: A scheme for pitch extraction of speech using autocorrelation function with frame length proportional to the time lag, Proc. IEEE ICASSP (1992) pp. 149–152

10.29 D.E. Terez: Robust pitch determination using nonlinear state-space embedding, Proc. IEEE ICASSP (2002)

10.30 C.M. Rader: Vector pitch detection, J. Acoust Soc. Am. **36**(C), 1463 (1964)

10.31 L.A. Yaggi: *Full Duplex Digital Vocoder* (Texas Instruments, Dallas 1962), Scientific Report No.1, SP14-A62; DDC-AD-282986

10.32 Y. Medan, E. Yair, D. Chazan: Super resolution pitch determination of speech signals, IEEE Trans. Signal Process. **39**, 40–48 (1991)

10.33 M.R. Weiss, R.P. Vogel, C.M. Harris: Implementation of a pitch-extractor of the double spectrum analysis type, J. Acoust. Soc. Am. **40**, 657–662 (1966)

10.34 H. Indefrey, W.J. Hess, G. Seeser: Design and evaluation of double-transform pitch determination algorithms with nonlinear distortion in the frequency domain, Proc. IEEE ICASSP, Vol. 2 (1985), paper 11.12

10.35 P. Martin: Comparison of pitch detection by cepstrum and spectral comb analysis, Proc. IEEE ICASSP (1982) pp. 180–183

10.36 V.T. Sreenivas: *Pitch estimation of aperiodic and noisy speech signals* (Indian Institute of Technology, Bombay 1982), Diss., Department of Electrical Engineering, Indian Institute of Technology

10.37 M.R. Schroeder: Period histogram and product spectrum: new methods for fundamental-frequency measurement, J. Acoust. Soc. Am. **43**, 819–834 (1968)

10.38 P. Martin: A logarithmic spectral comb method for fundamental frequency analysis, Proc. 11th Int. Congr. on Phonetic Sciences Tallinn (1987), paper 59.2

10.39 P. Martin: WinPitchPro – a tool for text to speech alignment and prosodic analysis, Proc. Speech Prosody 2004 (2004) pp. 545–548, http://www.isca-speech.org/archive/sp2004 and http://www.winpitch.com

10.40 J.C. Brown, M. Puckette: A high-resolution fundamental frequency determination based on phase changes of the Fourier transform, J. Acoust. Soc. Am. **94**, 662–667 (1993)

10.41 J.C. Brown: Musical fundamental frequency tracking using a pattern recognition method, J. Acoust. Soc. Am. **92**, 1394–1402 (1992)

10.42 F. Charpentier: Pitch detection using the short-term phase spectrum, Proc. IEEE ICASSP (1986) pp. 113–116

10.43 M. Lahat, R.J. Niederjohn, D.A. Krubsack: A spectral autocorrelation method for measurement of the fundamental frequency of noise-corrupted speech, IEEE Trans. Acoust. Speech Signal Process. **35**, 741–750 (1987)

10.44 B. Doval, X. Rodet: Estimation of fundamental frequency of musical sound signals, Proc. IEEE ICASSP (1991) pp. 3657–3660

10.45 T. Abe, K. Kobayashi, S. Imai: Robust pitch estimation with harmonics enhancement in noisy environments based on instantaneous frequency, Proc. ICSLP'96 (1996) pp. 1277–1280, http://www.isca-speech.org/archive/icslp_1996

10.46 T. Nakatani, T. Irino: Robust and accurate fundamental frequency estimation based on dominant harmonic components, J. Acoust. Soc. Am. **116**, 3690–3700 (2004)

10.47 L.R. Rabiner, M.J. Cheng, A.E. Rosenberg, C.A. McGonegal: A comparative study of several pitch detection algorithms, IEEE Trans. Acoust. Speech **24**, 399–423 (1976)

10.48 L. Arévalo: *Beiträge zur Schätzung der Frequenz gestörter Schwingungen kurzer Dauer und eine Anwendung auf die Analyse von Sprachsignalen* (Ruhr-Universität, Bochum 1991), Diss. in German

10.49 A.M. Noll, A. Michael: Pitch determination of human speech by the harmonic product spectrum the harmonic sum spectrum and a maximum likelihood estimate, Symp. Comput. Process. Commun. **19**, 779–797 (1970), ed. by the Microwave Inst., New York: Univ. of Brooklyn Press

10.50 D.H. Friedman: Pseudo-maximum-likelihood speech pitch extraction, IEEE Trans. Acoust. Speech Signal Process. **25**, 213–221 (1977)

10.51 R.J. McAulay, T.F. Quatieri: Pitch estimation and voicing detection based on a sinusoidal speech model, Proc. IEEE ICASSP (1990) pp. 249–252

10.52 A. Moreno, J.A.R. Fonollosa: Pitch determination of noisy speech using higher order statistics, Proc. IEEE ICASSP (1992) pp. 133–136

10.53 B.B. Wells: Voiced/Unvoiced decision based on the bispectrum, Proc. IEEE ICASSP (1985) pp. 1589–1592

10.54 J. Tabrikian, S. Dubnov, Y. Dickalov: Speech enhancement by harmonic modeling via MAP pitch tracking, Proc. IEEE ICASSP (2002) pp. 3316–3319

10.55 S. Godsill, M. Davy: Bayesian harmonic models for musical pitch estimation and analysis, Proc. IEEE ICASSP (2002) pp. 1769–1772

10.56 C.A. McGonegal, L.R. Rabiner, A.E. Rosenberg: A subjective evaluation of pitch detection methods using LPC synthesized speech, IEEE Trans. Acoust. Speech Signal Process. **25**, 221–229 (1977)

10.57 C. Hamon, E. Moulines, F. Charpentier: A diphone synthesis system based on time-domain prosodic modifications of speech, Proc. IEEE ICASSP (1989) pp. 238–241

10.58 D.M. Howard: Peak-picking fundamental period estimation for hearing prostheses, J. Acoust. Soc. Am. **86**, 902–910 (1989)

10.59 I. Dologlou, G. Carayannis: Pitch detection based on zero-phase filtering, Speech Commun. **8**, 309–318 (1989)

10.60 W.J. Hess: An algorithm for digital time-domain pitch period determination of speech signals and its application to detect F0 dynamics in VCV utterances, Proc. IEEE ICASSP (1976) pp. 322–325

10.61 T.V. Ananthapadmanabha, B. Yegnanarayana: Epoch extraction of voiced speech, IEEE Trans. Acoust. Speech Signal Process. **23**, 562–569 (1975)

10.62 L.O. Dolanský: An instantaneous pitch-period indicator, J. Acoust. Soc. Am. **27**, 67–72 (1955)

10.63 I.S. Howard, J.R. Walliker: The implementation of a portable real-time multilayer-perceptron speech fundamental period estimator, Proc. EUROSPEECH-89 (1989) pp. 206–209, http://www.isca-speech.org/archive/eurospeech_1989

10.64 W.J. Hess: A pitch-synchronous digital feature extraction system for phonemic recognition of speech, IEEE Trans. Acoust. Speech Signal Process. **24**, 14–25 (1976)

10.65 A. Davis, S. Nordholm, R. Togneri: Statistical voice activity detection using low-variance spectrum estimation and an adaptive threshold, IEEE Trans. Audio Speech Lang. Process. **14**, 412–424 (2006)

10.66 L.J. Siegel, A.C. Bessey: Voiced/unvoiced/mixed excitation classification of speech, IEEE Trans. Acoust. Speech Signal Process. **30**, 451–461 (1982)

10.67 S. Ahmadi, A.S. Spanias: Cepstrum-based pitch detection using a new statistical V/UV classification algorithm, IEEE Trans. Speech Audio Process. **7**, 333–338 (1999)

10.68 B.M. Lobanov, M. Boris: Automatic discrimination of noisy and quasi periodic speech sounds by the phase plane method, Soviet Physics – Acoustics 16, 353–356 (1970) Original (in Russian) in Akusticheskiy Zhurnal **16**, 425–428 (1970)

10.69 E. Fisher, J. Tabrikian, S. Dubnov: Generalized likelihood ratio test for voiced-unvoiced decision in noisy speech using the harmonic model, IEEE Trans. Audio Speech Lang. Process. **14**, 502–510 (2006)

10.70 B.S. Atal, L.R. Rabiner: A pattern recognition approach to voiced-unvoiced-silence classification with applications to speech recognition, IEEE Trans. Acoust. Speech Signal Process. **24**, 201–212 (1976)

10.71 O. Fujimura: An approximation to voice aperiodicity, IEEE Trans. Acoust. Speech Signal Process. **16**, 68–72 (1968)

10.72 A.K. Krishnamurthy, D.G. Childers: Two-channel speech analysis, IEEE Trans. Acoust Speech Signal Process. **34**, 730–743 (1986)

10.73 K.N. Stevens, D.N. Kalikow, T.R. Willemain: A miniature accelerometer for detecting glottal waveforms and nasalization, J. Speech Hear. Res. **18**, 594–599 (1975)

10.74 V.R. Viswanathan, W.H. Russell: *Subjective and objective evaluation of pitch extractors for LPC and harmonic-deviations vocoders* (Bolt Beranek and Newman, Cambridge 1984), MA: Report No. 5726

10.75 A.J. Fourcin, E. Abberton: First applications of a new laryngograph, Med Biol Illust **21**, 172–182 (1971)

10.76 D.G. Childers, M. Hahn, J.N. Larar: Silent and voiced/Unvoiced/Mixed excitation (four-way) classification of speech, IEEE Trans. Acoust. Speech Signal Process. **37**, 1771–1774 (1989)

10.77 E. Mousset, W.A. Ainsworth, J.A.R. Fonollosa: A comparison of several recent methods of fundamental frequency and voicing decision estimation, Proc. ICSLP'96 (1996) pp. 1273–1276, http://www.isca-speech.org/archive/icslp_1996

10.78 D.A. Krubsack, R.J. Niederjohn: An autocorrelation pitch detector and voicing decision with confidence measures developed for noise-corrupted speech, IEEE Trans. Signal Process. **39**, 319–329 (1991)

10.79 Y. Xu, X. Sun: Maximum speed of pitch change and how it may relate to speech, J. Acoust. Soc. Am. **111**, 1399–1413 (2002)

10.80 B.G. Secrest, G.R. Doddington: Postprocessing techniques for voice pitch trackers, Proc. IEEE ICASSP (1982) pp. 172–175

10.81 F. Plante, G.F. Meyer, W.A. Ainsworth: A pitch extraction reference database, Proc. EUROSPEECH'95 (1995) pp. 837–840, http://www.isca-speech.org/archive/eurospeech_1995

10.82 H. Kawahara, H. Katayose, A. de Cheveigné, R.D. Patterson: Fixed point analysis of frequency to instantaneous frequency mapping for accurate estimation of F0 and periodicity, Proc. EUROSPEECH'99 (1999) pp. 2781–2784, http://www.isca-speech.org/archive/eurospeech_1999

10.83 L.R. Rabiner, M.R. Sambur, C.E. Schmidt: Applications of nonlinear smoothing algorithm to speech processing, IEEE Trans. Acoust. Speech Signal Process. **23**, 552–557 (1975)

10.84 P. Specker: A powerful postprocessing algorithm for time-domain pitch trackers, Proc. IEEE ICASSP (1984), paper 28B.2

10.85 F. Itakura: Minimum prediction residual applied to speech recognition, IEEE Trans. Acoust. Speech Signal Process. **23**, 67–72 (1975)

10.86 Y.R. Wang, I.J. Wong, T.C. Tsao: A statistical pitch detection algorithm, Proc. IEEE ICASSP (2002) pp. 357–360

10.87 Y. Sagisaka, N. Campbell, N. Higuchi (eds.): *Computing prosody. Computational models for processing spontaneous speech* (Springer, New York 1996)

10.88 P. Bagshaw: *Automatic prosodic analysis for computer aided pronunciation teaching* (Univ. of Edinburgh, Edinburgh 1993), PhD Thesis http://www.cstr.ed.ac.uk/projects/fda/Bagshaw_PhDThesis.pdf

10.89 R.J. Baken: *Clinical Measurement of Speech and Voice* (Taylor Francis, London 1987)

10.90 A. Askenfelt: Automatic notation of played music: The Visa project, Fontes Artis Musicae **26**, 109–120 (1979)

10.91 Y.M. Cheng, D. O'Shaughnessy: Automatic and reliable estimation of glottal closure instant and period, IEEE Trans. Acoust. Speech Signal Process. **37**, 1805–1815 (1989)

10.92 W.J. Hess: Determination of glottal excitation cycles in running speech, Phonetica **52**, 196–204 (1995)

10.93 W.J. Hess: Pitch determination of acoustic signals – an old problem and new challenges, Proc. 18th Intern. Congress on Acoustics, Kyoto (2004), paper Tu2.H.1

10.94 B. Yegnanarayana, R. Smits: A robust method for determining instants of major excitations in voiced speech, Proc. IEEE ICASSP (1995) pp. 776–779

10.95 M. Brookes, P.A. Naylor, J. Gudnason: A quantitative assessment of group delay methods for identifying glottal closures in voiced speech, IEEE Trans. Audio Speech Language Process. **14**, 456–466 (2006)

10.96 C.X. Ma, Y. Kamp, L.F. Willems: A Frobenius norm approach to glottal closure detection from the speech signal, IEEE Trans. Speech Audio Process. **2**, 258–265 (1994)

10.97 L. Du, Z. Hou: Determination of the instants of glottal closure from speech wave using wavelet transform, http://www.icspat.com/papers/329mfi.pdf
10.98 K.E. Barner: Colored L-ℓ filters and their application in speech pitch detection, IEEE Trans. Signal Process. **48**, 2601–2606 (2000)
10.99 J.L. Navarro-Mesa, I. Esquerra-Llucià: A time-frequency approach to epoch detection, Proc. EUROSPEECH'95 (1995) pp. 405–408, http://www.isca-speech.org/archive/eurospeech_1995
10.100 A. de Cheveigné: Separation of concurrent harmonic sounds: Fundamental frequency estimation and a time-domain cancellation model of auditory processing, J. Acoust. Soc. Am. **93**, 3279–3290 (1993)
10.101 A.P. Klapuri: *Signal processing methods for the automatic transcription of music* (Tampere Univ. Technol., Tampere 2004), Ph.D. diss. http://www.cs.tut.fi/sgn/arg/klap/klap_phd.pdf
10.102 M. Goto: A predominant F0-estimation method for polyphonic musical audio signals, Proc. 18th Intern. Congress on Acoustics Kyoto (2004), paper Tu2.H.4
10.103 T. Tolonen, M. Karjalainen: A computationally efficient multipitch analysis model, IEEE Trans. Speech Audio Process. **8**, 708–716 (2000)
10.104 H. Kameoka, T. Nishimoto, S. Sagayama: Separation of harmonic structures based on tied Gaussian mixture model and information criterion for concurrent sounds, Proc. IEEE ICASSP (2004), paper AE-P5.9
10.105 A.P. Dempster, N.M. Laird, D.B. Rubin: Maximum likelihood from incomplete data via the EM algorithm, J. Roy. Stat. Soc. B **39**, 1–38 (1977)
10.106 L. Yoo, I. Fujinaga: A comparative latency study of hardware and software pitch-trackers, Proc. 1999 Int. Computer Music Conf. (1999) pp. 36–40

8. The Kalman Filter

S. Gannot, A. Yeredor

The Kalman filter and its variants are some of the most popular tools in statistical signal processing and estimation theory. In this chapter, we introduce the Kalman filter, providing a succinct, yet rigorous derivation thereof, which is based on the orthogonality principle. We also introduce several important variants of the Kalman filter, namely various Kalman smoothers, a Kalman predictor, a nonlinear extension (the extended Kalman filter), and adaptation to cases of temporally correlated measurement noise.

The application of the Kalman filter to two important speech processing problems, namely, speech enhancement and speaker localization is demonstrated.

The Kalman filter, together with its basic variants, are some of the most widely applied tools in fields related to statistical signal processing, especially in the context of causal, (nearly) real-time applications. In its fundamental, classical form, Kalman filtering is aimed at sequential (recursive) linear estimation of the state of a linear dynamic system from noisy measurements, linearly related to the unobserved state variables. The general statistical framework of the Kalman filter is Bayesian, namely a priori knowledge of statistical properties (up to second order) of the underlying process is assumed. This knowledge is specified through the mean and covariance of the initial state, and through the recursive model equation, which can be used to propagate these properties through time.

At each time instant, the Kalman filter's output is the optimal *linear* minimum mean square error (MSE) estimator of the state from measurements up to that time instant; moreover, it is the optimal minimum MSE estimator in general (i. e., among all linear and nonlinear estimators), when the both the states and the measurements share a jointly Gaussian probability distribution, specified by the model equations and by the measurement equations. The recursive structure of the filter (dwelling on the recursive structure of the model and measurement equations) is computationally appealing, as it enables to automatically (and optimally) take all past measurements into account, without the need to explicitly remember (namely, spend memory resources on) and account for these measurements. As an important byproduct, the Kalman filter equations also provide expressions for the MSE in the resulting estimates.

The Kalman filter has found numerous applications in fields related to control of dynamic systems. It is also used for estimating and predicting the trajectories of

moving objects, ranging from celestial bodies and missiles to microscopic particles. Since the late 1980s the Kalman filter was adopted for the estimation of speech signals contaminated by additive noise.

Rudolf Emil Kalman was born on 19 May 1930 in Budapest, Hungary (These historical notes are based on *Grewal* and *Andrews*' book [8.1]). During the second world war his family emigrated to the USA. In November 1958, while working in the Research Institute for Advanced Studies (RIAS), Kalman came up with the fundamental idea of applying state-space notation to the Wiener filter problem. This turned out to be the basis of the celebrated Kalman filter [8.2, 3]. The first known application of the Kalman filter, a trajectory estimation for the Apollo mission, was the work of *Schmidt* [8.4, 5] conducted at the Ames Research Center of the National Aeronautics and Space Administration (NASA). Since then the Kalman filter is an essential part of nearly every trajectory estimation task.

The Kalman filter and its variants (predictor, smoother and nonlinear forms) are widely addressed in the statistical signal processing and control theory literature. There is a plethora of articles and text books on both theoretical and practical aspects. The interested reader is referred to the books [8.1, 6, 7] and [8.8], to name just a few. A survey of associated speech-related papers can be found in Sect. 8.4.

This chapter is aimed at providing an overview of the basic Kalman filter, as well as some of its most popular variants, including concise, yet rigorous derivations of the resulting expressions. In addition, it provides emphasis on the applicability of Kalman filtering in speech-processing-related problems, such as speech enhancement (denoising) and speaker tracking. Although the general framework of Kalman filtering allows for complex-valued state vectors and measurements, we chose to concentrate, for simplicity, on the real-valued case, which is most commonly encountered in speech-related applications.

This chapter is structured as follows. In the next section, we provide the formulation of the estimation problem, as well as a rigorous derivation of the basic Kalman filter equations. In Sect. 8.2, we demonstrate the use of the Kalman filter in common stochastic denoising problems. Section 8.3 offers an overview of classical, immediate extensions of the Kalman filter, namely the Kalman predictor, the Kalman smoother (in various forms), and the extended Kalman filter (which addresses nonlinear models). Section 8.4 is dedicated to applications of Kalman filter in the context of speech processing; it provides a thorough literature survey of recent research activities in this context, as well as detailed examples of two popular applications, namely speech enhancement and speaker tracking.

8.1 Derivation of the Kalman Filter

Several approaches can be taken in the derivation of the Kalman filter. Following specification of the model equations and measurement equations, one can assume a Gaussian distribution of the driving processes, as well as of the initial state, and then derive the posterior distributions of the states given the observations, taking the mean of the resulting distributions as the states' estimates, namely as the filter's outputs. Another option is to take a recursive weighted least-squares (WLS) approach combined with special weighting of the previous estimate of the states in the role of additional *measurements*. Both of these approaches, while perfectly legitimate, tend to be somewhat cumbersome in the exposition of the associated derivations.

In this section, we will take a slightly different approach, dwelling on the well-known orthogonality principle, which is a basic property of the optimal linear estimator's estimation error. By showing that the associated optimality properties are automatically inherited from one recursion phase to the next, we would establish the claimed optimality of the entire scheme.

In the following subsection, we will review the optimal linear estimator (in the Bayesian framework) and the associated necessary and sufficient conditions for optimality. We will then exploit these conditions in the derivation of the Kalman filter in the subsequent subsection.

8.1.1 The Minimum Mean Square Linear Optimal Estimator

Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be two random vectors with an arbitrary distribution having finite moments up to second order. Denoting by $\boldsymbol{z} = (\boldsymbol{x}^{\mathrm{T}} \ \boldsymbol{y}^{\mathrm{T}})^{\mathrm{T}}$ the concatenation of the two vectors into a single vector, the mean and covariance of $\boldsymbol{z}$ are then given by

$$\boldsymbol{\eta}_z = E(\boldsymbol{z}) = \begin{pmatrix} \boldsymbol{\eta}_x \\ \boldsymbol{\eta}_y \end{pmatrix} \tag{8.1a}$$

$$\mathbf{C}_{zz} = E\big[(z-\boldsymbol{\eta}_z)(z-\boldsymbol{\eta}_z)^{\mathrm{T}}\big] = \begin{pmatrix} \mathbf{C}_{xx} & \mathbf{C}_{xy} \\ \mathbf{C}_{yx} & \mathbf{C}_{yy} \end{pmatrix} \tag{8.1b}$$

with obvious partitions according to the respective dimensions, where $\boldsymbol{\eta}_x$ and $\boldsymbol{\eta}_y$ denote the means of $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively, $\mathbf{C}_{xx}$, $\mathbf{C}_{yy}$, denote the covariance matrices of $\boldsymbol{x}$, $\boldsymbol{y}$, respectively, and $\mathbf{C}_{xy} = \mathbf{C}_{yx}^{\mathrm{T}}$ denotes the covariance between $\boldsymbol{x}$ and $\boldsymbol{y}$.

Assume now that it is desired to estimate $\boldsymbol{x}$ from observation of a $\boldsymbol{y}$ using linear estimation of the form

$$\hat{\boldsymbol{x}} = \mathbf{A}\boldsymbol{y} + \boldsymbol{b}\,, \tag{8.2}$$

where $\hat{\boldsymbol{x}}$ denotes the estimate of $\boldsymbol{x}$, and $\mathbf{A}$ and $\boldsymbol{b}$ are some fixed matrix and vector, respectively, of the appropriate dimensions. Note that any linear estimator can be represented in such a form, and its properties are uniquely determined by $\mathbf{A}$ and $\boldsymbol{b}$.

Assume further that it is desired to find the linear estimator that attains the smallest (matrix) MSE among all linear estimators. More specifically, denoting

$$\boldsymbol{\epsilon} = \hat{\boldsymbol{x}} - \boldsymbol{x} \tag{8.3}$$

as the estimation error, the (matrix) MSE is defined as

$$\mathbf{P} = E(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^{\mathrm{T}})\,. \tag{8.4}$$

An estimator $\hat{\boldsymbol{x}}_1$ is said to attain a smaller (matrix) MSE than another estimator, say $\hat{\boldsymbol{x}}_2$, when the matrix $\mathbf{P}_1$ attained by $\hat{\boldsymbol{x}}_1$ is smaller than the matrix $\mathbf{P}_2$ attained by $\hat{\boldsymbol{x}}_2$, in the sense that the difference matrix $\mathbf{P}_2 - \mathbf{P}_1$ is positive (semi-)definite.

Let us find $\mathbf{A}$ and $\boldsymbol{b}$ which minimize the MSE. For any $\mathbf{A}$, $\boldsymbol{b}$, the estimation error is given by

$$\begin{aligned} \boldsymbol{\epsilon} &= \hat{\boldsymbol{x}} - \boldsymbol{x} = \mathbf{A}\boldsymbol{y} + \boldsymbol{b} - \boldsymbol{x} \\ &= \mathbf{A}(\boldsymbol{y} - \boldsymbol{\eta}_y) - (\boldsymbol{x} - \boldsymbol{\eta}_x) + \boldsymbol{c}\,, \end{aligned} \tag{8.5}$$

where $\boldsymbol{c}$ is a constant vector, defined as

$$\boldsymbol{c} = \boldsymbol{b} + \mathbf{A}\boldsymbol{\eta}_y - \boldsymbol{\eta}_x\,. \tag{8.6}$$

We thus obtain

$$\begin{aligned} \mathbf{P} &= E(\boldsymbol{\epsilon}\boldsymbol{\epsilon}^{\mathrm{T}}) = E\big[(\mathbf{A}(\boldsymbol{y} - \boldsymbol{\eta}_y) - (\boldsymbol{x} - \boldsymbol{\eta}_x) + \boldsymbol{c}) \\ &\quad (\mathbf{A}(\boldsymbol{y} - \boldsymbol{\eta}_y) - (\boldsymbol{x} - \boldsymbol{\eta}_x) + \boldsymbol{c})^{\mathrm{T}}\big] \\ &= \mathbf{A}\mathbf{C}_{yy}\mathbf{A}^{\mathrm{T}} - \mathbf{A}\mathbf{C}_{yx} - \mathbf{C}_{xy}\mathbf{A}^{\mathrm{T}} + \mathbf{C}_{xx} + \boldsymbol{c}\boldsymbol{c}^{\mathrm{T}}\,. \end{aligned} \tag{8.7}$$

Since the term $\boldsymbol{c}\boldsymbol{c}^{\mathrm{T}}$ is positive semidefinite, $\mathbf{P}$ would definitely be minimized with respect to $\boldsymbol{c}$ (controlled by $\boldsymbol{b}$) by setting $\boldsymbol{c} = \mathbf{0}$, obtained by choosing

$$\boldsymbol{b} = \boldsymbol{\eta}_x - \mathbf{A}\boldsymbol{\eta}_y\,. \tag{8.8}$$

It now remains to minimize

$$\mathbf{P} = \mathbf{A}\mathbf{C}_{yy}\mathbf{A}^{\mathrm{T}} - \mathbf{A}\mathbf{C}_{yx} - \mathbf{C}_{xy}\mathbf{A}^{\mathrm{T}} + \mathbf{C}_{xx} \tag{8.9}$$

with respect to $\mathbf{A}$. Assuming that $\mathbf{C}_{yy}$ is invertible, we note that $\mathbf{P}$ in (8.9) can also be written as

$$\begin{aligned} \mathbf{P} &= \big(\mathbf{A} - \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\big)\mathbf{C}_{yy} \cdot \big(\mathbf{A} - \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\big)^{\mathrm{T}} \\ &\quad + \mathbf{C}_{xx} - \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx}\,. \end{aligned} \tag{8.10}$$

If $\mathbf{C}_{yy}$ is singular, this means that the measurements vector $\boldsymbol{y}$ contains some redundancy, as one or more linear combinations of its elements are zero (in the mean-squared sense). This means in turn, that $\boldsymbol{y}$ can be reduced into a smaller measurements vector with non-singular covariance, without loss in the attainable MSE in estimating $\boldsymbol{x}$. The first term in (8.10) is again positive semi-definite, and may be set to zero by selecting

$$\mathbf{A} = \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\,. \tag{8.11}$$

Since the other two terms do not depend on $\mathbf{A}$ (or on $\boldsymbol{b}$), this is the global minimizing solution, and the remainder is the residual (minimal) MSE matrix. Combining (8.8) and (8.11), we obtain the optimal linear estimator

$$\hat{\boldsymbol{x}} = \boldsymbol{\eta}_x + \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}(\boldsymbol{y} - \boldsymbol{\eta}_y)\,, \tag{8.12}$$

whose MSE is given by (8.10):

$$\mathbf{P} = \mathbf{C}_{xx} - \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx}\,. \tag{8.13}$$

8.1.2 The Estimation Error: Necessary and Sufficient Conditions for Optimality

Recall the estimation error (8.5) of any linear estimator:

$$\boldsymbol{\epsilon} = \mathbf{A}(\boldsymbol{y} - \boldsymbol{\eta}_y) - (\boldsymbol{x} - \boldsymbol{\eta}_x) + \boldsymbol{c}\,. \tag{8.14}$$

We obviously have

$$E(\boldsymbol{\epsilon}) = \boldsymbol{c} \tag{8.15}$$

and

$$E(\boldsymbol{\epsilon}\boldsymbol{y}^{\mathrm{T}}) = \mathbf{A}\mathbf{C}_{yy} - \mathbf{C}_{xy} + \boldsymbol{c}\boldsymbol{\eta}_y^{\mathrm{T}}\,. \tag{8.16}$$

When substituting (8.8) and (8.11) into (8.15) and (8.16) we easily obtain $E(\boldsymbol{\epsilon}) = \mathbf{0}$ and $E(\boldsymbol{\epsilon}\boldsymbol{y}^{\mathrm{T}}) = \mathbf{0}$. Moreover, it can be seen from (8.15) and (8.16) that, for any linear estimator, $E(\boldsymbol{\epsilon}) = \mathbf{0}$ and $E(\boldsymbol{\epsilon}\boldsymbol{y}^{\mathrm{T}}) = \mathbf{0}$ imply (respectively) (8.8) and (8.11), which in turn imply the MSE optimality of the estimator. We may therefore conclude with the following necessary and sufficient conditions for the optimal linear estimator.

Theorem 8.1

*Let $\hat{\boldsymbol{x}}$ denote an estimator of a random vector $\boldsymbol{x}$ from the random vector (*measurements*) $\boldsymbol{y}$, and let $\boldsymbol{\epsilon} = \hat{\boldsymbol{x}} - \boldsymbol{x}$ denote the estimation error. $\hat{\boldsymbol{x}}$ is the optimal linear estimator of $\boldsymbol{x}$ from $\boldsymbol{y}$ in the sense of minimum MSE if and only if the following three conditions hold:*

1. *$\hat{\boldsymbol{x}}$ is a linear function of $\boldsymbol{y}$,*
2. *$E(\boldsymbol{\epsilon}) = \mathbf{0}$,*
3. *$E(\boldsymbol{\epsilon}\boldsymbol{y}^T) = \mathbf{0}$.*

The condition of linearity is trivial. The zero-mean condition asserts that there is no constant bias in the optimal linear estimation, since any such bias can be removed in a linear operation, which would thereby reduce the MSE. The third condition asserts that there should be no correlation between any of the estimation error's elements and any of the measurements. This is a fundamental condition for MSE optimality of the optimal linear estimator: if any of its estimation error's elements were correlated with any of the measurements, then this would (intuitively) mean that by observing these measurements one could deduce the mean direction of departure of the respective errors from their (zero) mean. Such knowledge would in turn imply that a linear estimator of the respective estimation errors can be constructed and *appended* to the original estimator, so as to further reduce the MSE without breeching the linearity framework. If such an operation is possible, the original estimator cannot be MSE-optimal.

Therefore, the orthogonality condition, together with the zero-mean and linearity conditions, imply that all linear operations that may potentially reduce the MSE have been exhausted, and no further linear operations may be able to reduce the MSE further. Naturally, this implies MSE optimality of the respective linear estimator. In the sequel, we shall exploit these three conditions in constructively designing the estimator to satisfy all three, thereby inducing its optimality.

8.1.3 The Kalman Filter

Quite commonly, it is required to estimate time-dependent random vectors $\boldsymbol{x}_n$ from measurements $\boldsymbol{y}_n$ obtained sequentially in time (where n is the time index). Although the expressions derived above for the linear estimator's parameters (the matrix $\mathbf{A}$ and vector $\boldsymbol{b}$) can always be used, the associated computation of covariance matrices (and their inversion) may become prohibitively computationally demanding in such applications, especially as the dimension of the accumulated measurements vector placed in a matrix $\mathbf{Y} = (\boldsymbol{y}_1\ \boldsymbol{y}_2\ \cdots\ \boldsymbol{y}_n)$ increases. Even if these matrices are computed offline, the required storage space has to grow with the increase in the dimension of $\boldsymbol{y}$, as more measurements become available.

Often, however, the propagation of the statistics of the process of interest (together with the measurements) can be deduced from a recursive description of the model. It would therefore be computationally appealing if it were possible to apply a recursive scheme, in which the optimal linear estimator takes a similar form, with simplified computation of the associated matrices. Such a recursive (linear, optimal) estimation scheme is known as Kalman filtering.

Assume that the underlying process of interest satisfies the following recursive *model equations*:

$$\boldsymbol{x}_n = \boldsymbol{\Phi}_n \boldsymbol{x}_{n-1} + \boldsymbol{w}_n, \quad n = 1, 2, \ldots, \tag{8.17}$$

where $\boldsymbol{\Phi}_n$ denotes the (known) transition matrix from the state at time instant $n-1$ to the state at time instant n and $\boldsymbol{w}_n$ denotes the stochastic *driving noise* at time instant n, assumed to be of zero mean with known covariance $\mathbf{Q}_n$. The initial conditions for this set of equation are the mean $\boldsymbol{m}_0 = E(\boldsymbol{x}_0)$ and covariance $\mathbf{P}_0 = E\left((\boldsymbol{x}_0 - \boldsymbol{m}_0)(\boldsymbol{x}_0 - \boldsymbol{m}_0)^{\mathrm{T}}\right)$ of the initial state $\boldsymbol{x}_0$.

Assume further that the measurements $\boldsymbol{y}_n$ are related to the states $\boldsymbol{x}_n$ via the following *measurement equations*:

$$\boldsymbol{y}_n = \mathbf{H}_n \boldsymbol{x}_n + \boldsymbol{v}_n, \quad n = 1, 2, \ldots, \tag{8.18}$$

where $\mathbf{H}_n$ denotes the (known) transformation matrix from the state $\boldsymbol{x}_n$ to the measurement $\boldsymbol{y}_n$, and $\boldsymbol{v}_n$ denotes the stochastic *measurement noise* at time instant n, assumed to be of zero mean with known covariance $\mathbf{R}_n$.

It is further assumed that the initial state $\boldsymbol{x}_0$, the driving noise $\boldsymbol{w}_n$, and the measurement noise $\boldsymbol{v}_n$ are all uncorrelated, namely:

$$\begin{aligned}
&E\left(\boldsymbol{x}_0 \boldsymbol{w}_n^{\mathrm{T}}\right) = \mathbf{0} && E\left(\boldsymbol{x}_0 \boldsymbol{v}_n^{\mathrm{T}}\right) = \mathbf{0} && \forall n \\
&E\left(\boldsymbol{w}_n \boldsymbol{v}_m^{\mathrm{T}}\right) = \mathbf{0} && && \forall n, m \\
&E\left(\boldsymbol{w}_n \boldsymbol{w}_m^{\mathrm{T}}\right) = \mathbf{0} && E\left(\boldsymbol{v}_n \boldsymbol{v}_m^{\mathrm{T}}\right) = \mathbf{0} && \forall n \neq m \\
&E\left(\boldsymbol{w}_n \boldsymbol{w}_n^{\mathrm{T}}\right) = \mathbf{Q}_n && E\left(\boldsymbol{v}_n \boldsymbol{v}_n^{\mathrm{T}}\right) = \mathbf{R}_n\,. && \forall n
\end{aligned} \tag{8.19}$$

In the classical causal filtering framework, it is desired to compute, at each time instant n, the optimal linear estimator of $\boldsymbol{x}_n$ from all the measurements $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n$ up to the same time instant. We shall now propose a recursive scheme for this estimation process.

To this end, let us introduce a general simplified notation, denoting by $\hat{\boldsymbol{x}}_{k|\ell}$ the optimal linear estimator of $\boldsymbol{x}_k$ from $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_\ell$. Likewise, we denote by $\boldsymbol{\epsilon}_{k|\ell} = \hat{\boldsymbol{x}}_{k|\ell} - \boldsymbol{x}_k$ this estimator's estimation error, and by $\mathbf{P}_{k|\ell} = E(\boldsymbol{\epsilon}_{k|\ell}\boldsymbol{\epsilon}_{k|\ell}^{\mathrm{T}})$ its (matrix) MSE.

Assume now, that at time instant n we are given $\hat{\boldsymbol{x}}_{n-1|n-1}$. Let us proceed in the following two-step scheme:

1. Obtain $\hat{\boldsymbol{x}}_{n|n-1}$ from
$$\hat{\boldsymbol{x}}_{n|n-1} = \boldsymbol{\Phi}_n \hat{\boldsymbol{x}}_{n-1|n-1} \,, \tag{8.20}$$
2. Using the new measurement $\boldsymbol{y}_n$, obtain $\hat{\boldsymbol{x}}_{n|n}$ from
$$\hat{\boldsymbol{x}}_{n|n} = \hat{\boldsymbol{x}}_{n|n-1} + \mathbf{K}_n(\boldsymbol{y}_n - \mathbf{H}_n \hat{\boldsymbol{x}}_{n|n-1}) \,, \tag{8.21}$$
where $\mathbf{K}_n$ is a special *weighting matrix* to be derived shortly, often termed the *Kalman gain*.

These steps have the following intuitive interpretation:

- In the first step (often called the *propagation* step), we compute the estimate of $\boldsymbol{x}_n$ based on all the previous measurements up to $\boldsymbol{y}_{n-1}$. Since we already have the optimal estimate of $\boldsymbol{x}_{n-1}$ based on these measurements, the best we can do, in the absence of new information, is to apply the model equation to $\hat{\boldsymbol{x}}_{n-1|n-1}$, ignoring the unknown, zero-mean driving noise $\boldsymbol{w}_n$.
- In the second step (often called the *update* step), we use the new measurement $\boldsymbol{y}_n$ to update the estimate of $\boldsymbol{x}_n$ from the first step, which was only based on previous measurements. The update consists of an additive term, which is proportional to the *innovation* contained in the new measurement $\boldsymbol{y}_n$. This innovation reflects the *additional* information in $\boldsymbol{y}_n$ regarding the estimate $\hat{\boldsymbol{x}}_{n|n-1}$: if the measurement is equal to what could be expected from that estimate, namely to $\mathbf{H}_n\hat{\boldsymbol{x}}_{n|n-1}$, then there is no innovation (the term is zero), and consequently $\hat{\boldsymbol{x}}_{n|n-1}$ is deemed *good enough* to become $\hat{\boldsymbol{x}}_{n|n}$, having been *confirmed* by $\boldsymbol{y}_n$. However, usually this is not the case, and a nonzero innovation has to be incorporated when updating $\hat{\boldsymbol{x}}_{n|n-1}$ to $\hat{\boldsymbol{x}}_{n|n}$. The proper weighting in the transformation from the innovation to the update is prescribed by the Kalman gain matrix $\mathbf{K}_n$.

Natural and intuitively appealing as it may be, this explanation can by no means serve as a proof of optimality. We shall now derive a rigorous proof of optimality, based on the three necessary and sufficient conditions mentioned above, namely linearity, zero-mean of the error, and orthogonality of the error to all measurements on which the estimate is based. Our proof will be based on recursive induction of these conditions, dwelling on the recursive structure of the estimator. An important byproduct of the proof will be an expression for the Kalman gain matrix $\mathbf{K}_n$, obtained by enforcing the optimality conditions. We will also obtain important expressions for the associated error covariances $\mathbf{P}_{n|n-1}$ and $\mathbf{P}_{n|n}$.

As mentioned above, let us assume that at time instant n we have $\hat{\boldsymbol{x}}_{n-1|n-1}$, the optimal linear estimator of $\boldsymbol{x}_{n-1}$ from $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_{n-1}$. Its associated estimation error is $\boldsymbol{\epsilon}_{n-1|n-1}$, which by virtue of the optimality of $\hat{\boldsymbol{x}}_{n-1|n-1}$, has zero mean and is orthogonal to $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_{n-1}$.

We will now show that the proposed scheme propagates the optimality conditions from $\hat{\boldsymbol{x}}_{n-1|n-1}$ to the resulting estimate $\hat{\boldsymbol{x}}_{n|n}$ (and, also, along the way, to the intermediate estimate $\hat{\boldsymbol{x}}_{n|n-1}$). Considering the linearity condition first, it is obvious that, since $\hat{\boldsymbol{x}}_{n-1|n-1}$ is a linear function of all $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_{n-1}$, so is $\hat{\boldsymbol{x}}_{n|n-1}$ obtained from (8.20) and thus also $\hat{\boldsymbol{x}}_{n|n}$ obtained from (8.21).

To examine the two remaining conditions, let us monitor the propagation of the estimation error. Subtracting $\boldsymbol{x}_n$ from (8.20), we obtain, by substituting (8.17),

$$\begin{aligned}\boldsymbol{\epsilon}_{n|n-1} &= \hat{\boldsymbol{x}}_{n|n-1} - \boldsymbol{x}_n \\ &= \boldsymbol{\Phi}_n \hat{\boldsymbol{x}}_{n-1|n-1} - (\boldsymbol{\Phi}_n \boldsymbol{x}_{n-1} + \boldsymbol{w}_n) \\ &= \boldsymbol{\Phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1} - \boldsymbol{x}_{n-1}) - \boldsymbol{w}_n \\ &= \boldsymbol{\Phi}_n \boldsymbol{\epsilon}_{n-1|n-1} - \boldsymbol{w}_n \,. \end{aligned} \tag{8.22}$$

Likewise, subtracting $\boldsymbol{x}_n$ from (8.21), we get, by substituting (8.18),

$$\begin{aligned}\boldsymbol{\epsilon}_{n|n} &= \hat{\boldsymbol{x}}_{n|n} - \boldsymbol{x}_n \\ &= \hat{\boldsymbol{x}}_{n|n-1} + \mathbf{K}_n(\boldsymbol{y}_n - \mathbf{H}_n \hat{\boldsymbol{x}}_{n|n-1}) - \boldsymbol{x}_n \\ &= \boldsymbol{\epsilon}_{n|n-1} + \mathbf{K}_n(\mathbf{H}_n \boldsymbol{x}_n + \boldsymbol{v}_n - \mathbf{H}_n \hat{\boldsymbol{x}}_{n|n-1}) \\ &= \boldsymbol{\epsilon}_{n|n-1} + \mathbf{K}_n(-\mathbf{H}_n \boldsymbol{\epsilon}_{n|n-1} + \boldsymbol{v}_n) \\ &= (\mathbf{I} - \mathbf{K}_n \mathbf{H}_n)\boldsymbol{\epsilon}_{n|n-1} + \mathbf{K}_n \boldsymbol{v}_n \,. \end{aligned} \tag{8.23}$$

It is now easily observed that from (8.22)

$$E(\boldsymbol{\epsilon}_{n|n-1}) = \boldsymbol{\Phi}_n E(\boldsymbol{\epsilon}_{n-1|n-1}) - E(\boldsymbol{w}_n) = \mathbf{0} \,, \tag{8.24}$$

and hence from (8.23),

$$E(\boldsymbol{\epsilon}_{n|n}) = (\mathbf{I} - \mathbf{K}_n \mathbf{H}_n) E(\boldsymbol{\epsilon}_{n|n-1}) + \mathbf{K}_n E(\boldsymbol{v}_n) = \mathbf{0} \,, \tag{8.25}$$

thus the zero-mean condition is seen to propagate simply by construction of the proposed estimation scheme. It now remains to verify the orthogonality condition. Let

us first examine the orthogonality of $\boldsymbol{\epsilon}_{n|n-1}$ to all the measurements on which $\hat{\boldsymbol{x}}_{n|n-1}$ is based, namely to all $\boldsymbol{y}_\ell$, $\ell = 1, 2, \ldots, n-1$. Using (8.22) again we get

$$\begin{aligned} E\big(\boldsymbol{\epsilon}_{n|n-1}\boldsymbol{y}_\ell^{\mathrm{T}}\big) &= E\big[(\boldsymbol{\Phi}_n\boldsymbol{\epsilon}_{n-1|n-1} - \boldsymbol{w}_n)\boldsymbol{y}_\ell^{\mathrm{T}}\big] \\ &= \boldsymbol{\Phi}_n E\big(\boldsymbol{\epsilon}_{n-1|n-1}\boldsymbol{y}_\ell^{\mathrm{T}}\big) - E\big(\boldsymbol{w}_n\boldsymbol{y}_\ell^{\mathrm{T}}\big) \\ &= \mathbf{0}\,, \qquad \ell = 1, 2, \ldots, n-1, \end{aligned} \tag{8.26}$$

where the first term is zeroed-out due to the optimality of $\hat{\boldsymbol{x}}_{n-1|n-1}$, and the second term is zeroed-out since all of the measurements $\boldsymbol{y}_\ell$ up to $\ell = n-1$ are essentially linear functions of the random vectors $\boldsymbol{x}_0$, $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{n-1}$, $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_{n-1}$, which are all orthogonal to $\boldsymbol{w}_n$ – thus these measurements are all orthogonal to $\boldsymbol{w}_n$ as well. Note that this already implies that the intermediate estimate $\hat{\boldsymbol{x}}_{n|n-1}$ of (8.20) is the optimal linear estimator of $\boldsymbol{x}_n$ from all measurements up to the previous time instant $n-1$.

We now wish to verify a similar orthogonality condition for $\boldsymbol{\epsilon}_{n|n}$. Note that for optimality of $\hat{\boldsymbol{x}}_{n|n}$, $\boldsymbol{\epsilon}_{n|n}$ has to be orthogonal to all $\boldsymbol{y}_\ell$ for $\ell = 1, 2, \ldots, n$, namely to the same measurements as $\boldsymbol{\epsilon}_{n|n-1}$, plus the new measurement $\boldsymbol{y}_n$ used in the update stage. Let us first examine the orthogonality to past measurements only, up to $\ell = n-1$. Using (8.23) we get

$$\begin{aligned} E\big(\boldsymbol{\epsilon}_{n|n}\boldsymbol{y}_\ell^{\mathrm{T}}\big) &= E\big\{[(\mathbf{I} - \mathbf{K}_n\mathbf{H}_n)\boldsymbol{\epsilon}_{n|n-1} + \mathbf{K}_n\boldsymbol{v}_n]\boldsymbol{y}_\ell^{\mathrm{T}}\big\} \\ &= (\mathbf{I} - \mathbf{K}_n\mathbf{H}_n)E\big(\boldsymbol{\epsilon}_{n|n-1}\boldsymbol{y}_\ell^{\mathrm{T}}\big) + \mathbf{K}_n E\big(\boldsymbol{v}_n\boldsymbol{y}_\ell^{\mathrm{T}}\big) \\ &= \mathbf{0}\,, \qquad \ell = 1, 2, \ldots, n-1, \end{aligned} \tag{8.27}$$

where the first term is zeroed-out following (8.26), and the second is zeroed-out due to a similar argument as used above for (8.26).

Note that all of the conditions up to this point were propagated *automatically* by the structure of the estimation scheme, regardless of the value of $\mathbf{K}_n$. By imposing the remaining condition of orthogonality of $\boldsymbol{\epsilon}_{n|n}$ to $\boldsymbol{y}_n$, we would now get an explicit expression for $\mathbf{K}_n$. We have

$$\begin{aligned} E\big(\boldsymbol{\epsilon}_{n|n}\boldsymbol{y}_n^{\mathrm{T}}\big) &= E\{[(\mathbf{I} - \mathbf{K}_n\mathbf{H}_n)\boldsymbol{\epsilon}_{n|n-1} + \mathbf{K}_n\boldsymbol{v}_n] \\ &\quad \cdot (\mathbf{H}_n\boldsymbol{x}_n + \boldsymbol{v}_n)^{\mathrm{T}}\} \\ &= E\big\{[(\mathbf{I} - \mathbf{K}_n\mathbf{H}_n)\boldsymbol{\epsilon}_{n|n-1} + \mathbf{K}_n\boldsymbol{v}_n] \\ &\quad \cdot (\mathbf{H}_n(\hat{\boldsymbol{x}}_{n|n-1} \pm \boldsymbol{\epsilon}_{n|n-1}) + \boldsymbol{v}_n)^{\mathrm{T}}\big\}\,. \end{aligned} \tag{8.28}$$

This expression involves six types of expectations that have to be computed, as follows:

$$E\big(\boldsymbol{\epsilon}_{n|n-1}\hat{\boldsymbol{x}}_{n|n-1}^{\mathrm{T}}\big) = \mathbf{0}\,, \tag{8.29a}$$

since $\hat{\boldsymbol{x}}_{n|n-1}$ is a linear function of measurements $\boldsymbol{y}_\ell$ up to $\ell = n-1$, all of which are orthogonal to $\boldsymbol{\epsilon}_{n|n-1}$ [by (8.26)];

$$E\big(\boldsymbol{\epsilon}_{n|n-1}\boldsymbol{\epsilon}_{n|n-1}^{\mathrm{T}}\big) = \mathbf{P}_{n|n-1}\,, \tag{8.29b}$$

by definition;

$$E\big(\boldsymbol{\epsilon}_{n|n-1}\boldsymbol{v}_n^{\mathrm{T}}\big) = \mathbf{0}\,, \tag{8.29c}$$

since $\boldsymbol{\epsilon}_{n|n-1}$ is a linear function of past measurements and of $\boldsymbol{x}_n$, all of which are in turn linear functions of $\boldsymbol{x}_0$, $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_n$ and $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_{n-1}$, which are all orthogonal to $\boldsymbol{v}_n$;

$$E\big(\boldsymbol{v}_n\hat{\boldsymbol{x}}_{n|n-1}^{\mathrm{T}}\big) = \mathbf{0}\,, \tag{8.29d}$$

due to a similar argument;

$$E\big(\boldsymbol{v}_n\boldsymbol{\epsilon}_{n|n-1}^{\mathrm{T}}\big) = \mathbf{0}\,, \tag{8.29e}$$

[same as (8.29c)]; and

$$E\big(\boldsymbol{v}_n\boldsymbol{v}_n^{\mathrm{T}}\big) = \mathbf{R}_n\,, \tag{8.29f}$$

from the measurement equation (8.18). To conclude, we obtain, substituting (8.29a)–(8.29f) into (8.28)

$$E\big(\boldsymbol{\epsilon}_{n|n}\boldsymbol{y}_n^{\mathrm{T}}\big) = (\mathbf{I} - \mathbf{K}_n\mathbf{H}_n)\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}} + \mathbf{K}_n\mathbf{R}_n\,, \tag{8.30}$$

which has to equal $\mathbf{0}$ to secure all optimality conditions. This leads to the requirement

$$\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}} = \mathbf{K}_n\big(\mathbf{H}_n\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}} + \mathbf{R}_n\big)\,, \tag{8.31}$$

which is easily satisfied by setting the Kalman gain $\mathbf{K}_n$ to be

$$\mathbf{K}_n = \mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}}\big(\mathbf{H}_n\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}} + \mathbf{R}_n\big)^{-1}\,. \tag{8.32}$$

A remaining practical issue, which is also of interest for performance evaluation, is the computation of the error covariance $\mathbf{P}_{n|n-1}$, required in the computation of (8.32). We take a similar recursive approach in the computation of these matrices: assume that along with $\hat{\boldsymbol{x}}_{n-1|n-1}$ (the previous estimate) we are also given $\mathbf{P}_{n-1|n-1}$. From (8.22) we have

$$\begin{aligned} &\mathbf{P}_{n|n-1} \\ &= E\big(\boldsymbol{\epsilon}_{n|n-1}\boldsymbol{\epsilon}_{n|n-1}^{\mathrm{T}}\big) \\ &= E\big[(\boldsymbol{\Phi}_n\boldsymbol{\epsilon}_{n-1|n-1} - \boldsymbol{w}_n)(\boldsymbol{\Phi}_n\boldsymbol{\epsilon}_{n-1|n-1} - \boldsymbol{w}_n)^{\mathrm{T}}\big] \\ &= \boldsymbol{\Phi}_n\mathbf{P}_{n-1|n-1}\boldsymbol{\Phi}_n^{\mathrm{T}} + \mathbf{Q}_n\,, \end{aligned} \tag{8.33}$$

in which we used the orthogonality of $\boldsymbol{w}_n$ to $\boldsymbol{\epsilon}_{n-1|n-1}$, the latter being a linear function of random vectors from the past, which are all uncorrelated with $\boldsymbol{w}_n$.

Although $\mathbf{P}_{n|n-1}$ is sufficient for computing $\mathbf{K}_n$, we are also interested in $\mathbf{P}_{n|n}$, both for obtaining the estimate's MSE and for passing that matrix onto the following recursion phase. From (8.23) we get

$$\begin{aligned}\mathbf{P}_{n|n} &= E\big(\boldsymbol{\epsilon}_{n|n}\boldsymbol{\epsilon}_{n|n}^{\mathrm{T}}\big)\\ &= E\big\{[(\mathbf{I}-\mathbf{K}_n\mathbf{H}_n)\boldsymbol{\epsilon}_{n|n-1}+\mathbf{K}_n\boldsymbol{v}_n]\\ &\quad\cdot(\mathbf{I}-\mathbf{K}_n\mathbf{H}_n)\boldsymbol{\epsilon}_{n|n-1}+\mathbf{K}_n\boldsymbol{v}_n)^{\mathrm{T}}\big\}\\ &= (\mathbf{I}-\mathbf{K}_n\mathbf{H}_n)\mathbf{P}_{n|n-1}(\mathbf{I}-\mathbf{K}_n\mathbf{H}_n)\\ &\quad+\mathbf{K}_n\mathbf{R}_n\mathbf{K}_n^{\mathrm{T}}\,, \end{aligned} \tag{8.34}$$

again having used the orthogonality of $\boldsymbol{v}_n$ to $\boldsymbol{\epsilon}_{n|n-1}$, a linear function of past measurements and of $\boldsymbol{x}_n$, which are all orthogonal to $\boldsymbol{v}_n$. Although (8.34) can indeed serve to compute $\mathbf{P}_{n|n}$, considerable simplification of this expression is possible. In fact, it can be shown that this expression is equivalent to the following:

$$\mathbf{P}_{n|n} = (\mathbf{I}-\mathbf{K}_n\mathbf{H}_n)\mathbf{P}_{n|n-1}\,. \tag{8.35}$$

To show this, observe that the difference between (8.34) and (8.35) can be expressed as

$$\begin{aligned}&-(\mathbf{I}-\mathbf{K}_n\mathbf{H}_n)\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}}\mathbf{K}_n^{\mathrm{T}}+\mathbf{K}_n\mathbf{R}_n\mathbf{K}_n^{\mathrm{T}}\\ &=\big(-\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}}+\mathbf{K}_n\mathbf{H}_n\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}}+\mathbf{K}_n\mathbf{R}_n\big)\mathbf{K}_n^{\mathrm{T}}\\ &=\big[-\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}}+\mathbf{K}_n\big(\mathbf{H}_n\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}}+\mathbf{R}_n\big)\big]\mathbf{K}_n^{\mathrm{T}}\,. \end{aligned} \tag{8.36}$$

Substituting (8.31) we easily observe that this difference vanishes, implying the equivalence of the two expressions.

This concludes the derivation of the expressions necessary to carry out a single recursion phase, in which the optimal linear estimate and its MSE matrix, provided from the previous recursion phase, are used, together with a new measurement, to compute an updated optimal linear estimate and its associated MSE matrix. In such a recursive structure, the output of each recursion phase serves as the input to the next recursion phase, and optimality is maintained throughout. A small remaining issue is the initialization of the process: we first need $\hat{\boldsymbol{x}}_{0|0}$, the MSE-optimal linear estimate of $\boldsymbol{x}_0$ based on no measurements at all. Naturally, that MSE-optimal estimator is the known mean of $\boldsymbol{x}_0$, namely $\boldsymbol{m}_0$. Moreover, the required covariance of that estimate is given by

$$\begin{aligned}\mathbf{P}_{0|0} &= E\big[(\hat{\boldsymbol{x}}_{0|0}-\boldsymbol{x}_0)(\hat{\boldsymbol{x}}_{0|0}-\boldsymbol{x}_0)^{\mathrm{T}}\big]\\ &= E\big[(\boldsymbol{m}_0-\boldsymbol{x}_0)(\boldsymbol{m}_0-\boldsymbol{x}_0)^{\mathrm{T}}\big]=\mathbf{P}_0\,. \end{aligned} \tag{8.37}$$

We are now ready to summarize the recipe for applying the Kalman filter:
Initialization:

$$\begin{aligned}&\text{Let } \hat{\boldsymbol{x}}_{0|0} := \boldsymbol{m}_0\,,\\ &\text{and } \mathbf{P}_{0|0} := \mathbf{P}_0\,.\end{aligned}$$

Proceed for n=1,2, . . . :
Propagation equations:

$$\hat{\boldsymbol{x}}_{n|n-1} := \boldsymbol{\Phi}_n\hat{\boldsymbol{x}}_{n-1|n-1}\,, \tag{p1}$$

$$\mathbf{P}_{n|n-1} := \boldsymbol{\Phi}_n\mathbf{P}_{n-1|n-1}\boldsymbol{\Phi}_n^{\mathrm{T}}+\mathbf{Q}_n\,. \tag{p2}$$

Update equations:

$$\mathbf{K}_n := \mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}}\big(\mathbf{H}_n\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}}+\mathbf{R}_n\big)^{-1}\,, \tag{u1}$$

$$\hat{\boldsymbol{x}}_{n|n} := \hat{\boldsymbol{x}}_{n|n-1}+\mathbf{K}_n(\boldsymbol{y}_n-\mathbf{H}_n\hat{\boldsymbol{x}}_{n|n-1})\,, \tag{u2}$$

$$\mathbf{P}_{n|n} := (\mathbf{I}-\mathbf{K}_n\mathbf{H}_n)\mathbf{P}_{n|n-1}\,. \tag{u3}$$

Note that (p2), (u1), and (u3) are all data independent, so wherever applicable, they can be computed in advance before the data sequence arrives. Only (p1) and (u2) need to be computed online. This appealing feature not only enables to save real-time computations, but also enables one to assess the propagation of MSE in time beforehand, as this MSE, expressed by the sequence of $\mathbf{P}_{n|n}$, is data independent.

8.2 Examples: Estimation of Parametric Stochastic Process from Noisy Observations

Formulating the Kalman filter for the estimation of a stochastic process contaminated by additive noise often boils down to the definition of the state-space equations. In this section, we will exemplify the procedure for noisy observations of several quite common stochastic processes. Although we shall consider stationary processes, the derivation of the Kalman filter does not necessitate any stationarity assumption. In Sect. 8.4 we will consider speech signals, which can only be assumed to be quasistationary.

Consider a difference equation with time-invariant coefficients,

$$x(n) = -\sum_{k=1}^{p}\alpha_k x(n-k)+\sum_{k=0}^{q}\beta_k w(n-k), \tag{8.38}$$

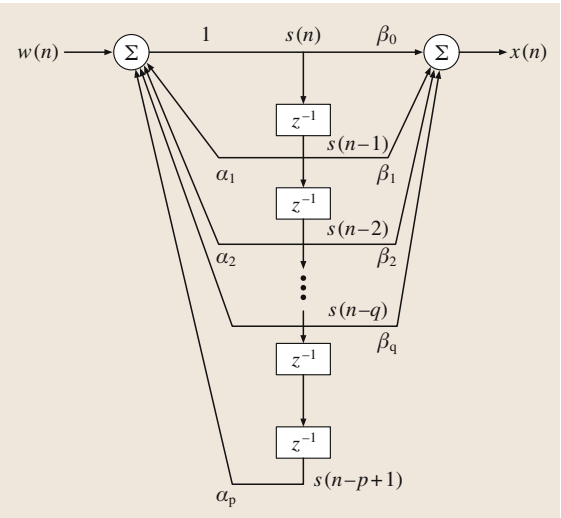


Fig. 8.1 A block diagram presenting an autoregressive moving-average process

where $w(n)$ is some zero-mean white-noise input, with variance $E[w^2(n)] = \sigma_w^2$. The output signal is usually termed an autoregressive moving-average (ARMA) process. The process $x(n)$ is a sum of two terms. The first term is a weighted sum of past output signal values [hence the term autoregressive (AR)]. The time-invariant AR coefficients, $\alpha_1, \ldots, \alpha_p$ are the weight values. The second term is a weighted average of past and current input samples [hence the term moving-average (MA)]. The time-invariant MA coefficients, $\beta_0, \ldots, \beta_q$, are the weight values. A possible block diagram of the process generation is depicted in Fig. 8.1; this form is often called *direct form II*, and is one of several alternative forms.

The block z^{-1} denotes a unit delay. It is desired to estimate $x(n)$ from noisy observations thereof, $y(n)$, given by

$$y(n) = x(n) + v(n), \tag{8.39}$$

where $v(n)$ denotes additive background noise with variance σ_v^2.

We will now turn to a formulation of the state-space representation of three special cases of the difference equation given in (8.38), namely, AR, MA, and ARMA processes. The state variables will be defined, in all three cases, as the inputs of the delay units, constituting the $\max(p, q+1)\times 1$ state vector

$$\boldsymbol{x}_n^{\mathrm{T}} = (s(n-p+1)\; s(n-p+2)\; \ldots\; s(n)) \,. \tag{8.40}$$

We assume without loss of generality that $p > q$. If this inequality does not hold, extra, zero-valued coefficients can be appended to the existing coefficients.

8.2.1 Autoregressive (AR) Process

In the autoregressive case, the coefficients $\beta_1, \ldots, \beta_q$ are assumed to be zero, while $\beta_0 = 1$. The resulting process is given by

$$x(n) = -\sum_{k=1}^{p} \alpha_k x(n-k) + w(n) \,. \tag{8.41}$$

The $p\times 1$ state-vector is given by (8.40), where the state variable $s(n)$ is essentially the output signal $x(n)$. In some cases it might be useful to define a longer state-vector with $p+1$ elements $x(n-p), \ldots, x(n)$. We will elaborate on such a case when discussing the application to speech signals in Sect. 8.4. However, continuing with the $p\times 1$ formulation, define the $p\times p$ transition matrix

$$\boldsymbol{\Phi} = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 1 \\ -\alpha_p & -\alpha_{p-1} & \cdots & -\alpha_2 & -\alpha_1 \end{pmatrix} . \tag{8.42}$$

Note that the transition matrix is time invariant in this case. Define also the vector $(p\times 1)$ and matrix $(1\times p)$

$$\boldsymbol{w}_n^{\mathrm{T}} = (0\; \ldots\; 0\; w(n)) \,, \tag{8.43}$$

$$\mathbf{H} = (0\; \ldots\; 0\; 1) \,. \tag{8.44}$$

Then (8.41) and (8.39) can be rewritten as

$$\begin{aligned} \boldsymbol{x}_n &= \boldsymbol{\Phi}\boldsymbol{x}_{n-1} + \boldsymbol{w}_n \\ y(n) &= \mathbf{H}\boldsymbol{x}_n + v(n) \,. \end{aligned} \tag{8.45}$$

Accordingly, the driving noise correlation matrix is given by

$$\mathbf{Q} = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & \sigma_w^2 \end{pmatrix} \tag{8.46}$$

and, since in this case the measurement vector is a scalar, the measurement noise correlation matrix is the scalar $\mathbf{R} = \sigma_v^2$. Using the state-space representation in (8.45) the Kalman filter is readily applied.

8.2.2 Moving-Average (MA) Process

In the moving-average case, the coefficients $\alpha_1, \ldots, \alpha_p$ are assumed to be identically zero, hence the process model simplifies to

$$x(n) = \sum_{k=0}^{q} \beta_k w(n-k) \,. \tag{8.47}$$

The $(q+1)\times 1$ state-vector is accordingly given in a form similar to (8.40) with p replaced by $q+1$. Now the state variable $s(n)$ is essentially the input signal $w(n)$. Using the above definitions, we can reformulate (8.47) in a state-space representation. The $(q+1)\times(q+1)$ transition matrix becomes a simple time-shift matrix,

$$\boldsymbol{\Phi} = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix} . \tag{8.48}$$

Define the vector $[(q+1)\times 1]$ and matrix $[1\times(q+1)]$

$$\boldsymbol{w}_n^{\mathrm{T}} = (0 \ \ldots \ 0 \ w(n)) \,, \tag{8.49}$$

$$\mathbf{H} = (\beta_q \ \ldots \ \beta_1 \ \beta_0) \,. \tag{8.50}$$

Then (8.47) and (8.39) can again be rewritten as

$$\begin{aligned} \boldsymbol{x}_n &= \boldsymbol{\Phi}\boldsymbol{x}_{n-1} + \boldsymbol{w}_n \\ y(n) &= \mathbf{H}\boldsymbol{x}_n + v(n) \,. \end{aligned} \tag{8.51}$$

As in the previous case (the AR process), the driving noise correlation matrix is given by (8.46) and the measurement noise correlation matrix by the scalar $\mathbf{R} = \sigma_v^2$. The state-space representation (8.51) can be used in the Kalman formulation.

8.2.3 Autoregressive Moving-Average (ARMA) Process

The state vector of the ARMA process is also given by (8.40) (assuming $p > q$), but now it is not directly related to either the input or the output signals. The transition matrix is identical to the $p\times p$ AR transition matrix in (8.42). The $p\times 1$ measurement matrix is equal to the corresponding MA measurement matrix padded with $p-q-1$ zeros

$$\mathbf{H} = (\beta_q \ \ldots \ \beta_1 \ \beta_0 \ 0 \ \ldots \ 0) \,. \tag{8.52}$$

The driving noise vector is identical to both the AR and MA processes and is given by (8.43). As in the special cases, the driving noise correlation matrix is given by (8.46) and the measurement noise correlation matrix is the scalar $\mathbf{R} = \sigma_v^2$.

8.2.4 The Case of Temporally Correlated Noise

The assumption that the measurement noise is temporally uncorrelated, i. e., $E(\boldsymbol{v}_n \boldsymbol{v}_m^{\mathrm{T}}) = \mathbf{0}$, $\forall n \neq m$ is not always realistic. To circumvent the need to account explicitly for correlated noise in the Kalman filter equations, the following procedure is commonly used.

Assume that the measurement noise $\boldsymbol{v}_n$ can be described using a state-space model as well. Let $\boldsymbol{x}_n^{\mathrm{v}}$ denote the respective noise state vector. Then

$$\boldsymbol{x}_n^{\mathrm{v}} = \boldsymbol{\Phi}_n^{\mathrm{v}} \boldsymbol{x}_{n-1}^{\mathrm{v}} + \boldsymbol{w}_n^{\mathrm{v}} \,, \quad n = 1, 2, \ldots \,, \tag{8.53}$$

where $\boldsymbol{\Phi}_n^{\mathrm{v}}$ is a known matrix, and $\boldsymbol{w}_n^{\mathrm{v}}$ is a temporally uncorrelated zero-mean driving noise (for the noise signal) with correlation matrix $E[\boldsymbol{w}_n^{\mathrm{v}}(\boldsymbol{w}_n^{\mathrm{v}})^{\mathrm{T}}] = \mathbf{Q}_n^{\mathrm{v}}$. The measurement noise $\boldsymbol{v}_n$ is related to the noise state vector $\boldsymbol{x}_n^{\mathrm{v}}$ by:

$$\boldsymbol{v}_n = \mathbf{H}_n^{\mathrm{v}} \boldsymbol{x}_n^{\mathrm{v}} \,, \quad n = 1, 2, \ldots \,, \tag{8.54}$$

where $\mathbf{H}_n^{\mathrm{v}}$ is known.

Assume that the recursive model in (8.17) and the measurement equations (8.18) still hold. We restate these equation with a slight change of notations. The signal state-space model is given by:

$$\boldsymbol{x}_n^{\mathrm{s}} = \boldsymbol{\Phi}_n^{\mathrm{s}} \boldsymbol{x}_{n-1}^{\mathrm{s}} + \boldsymbol{w}_n^{\mathrm{s}} \quad n = 1, 2, \ldots \tag{8.55}$$

and the signal measurement model is given by:

$$\boldsymbol{y}_n = \mathbf{H}_n^{\mathrm{s}} \boldsymbol{x}_n^{\mathrm{s}} + \boldsymbol{v}_n = \mathbf{H}_n^{\mathrm{s}} \boldsymbol{x}_n^{\mathrm{s}} + \mathbf{H}_n^{\mathrm{v}} \boldsymbol{x}_n^{\mathrm{v}} \,, \quad n = 1, 2, \ldots \tag{8.56}$$

Therefore, concatenating both state vectors will enable the use of the *standard* Kalman filter. Let

$$\boldsymbol{x}_n^{\mathrm{T}} = \left((\boldsymbol{x}_n^{\mathrm{s}})^{\mathrm{T}} \ (\boldsymbol{x}_n^{\mathrm{v}})^{\mathrm{T}} \right)$$

be the augmented state vector, and define

$$\boldsymbol{w}_n^{\mathrm{T}} = \left((\boldsymbol{w}_n^{\mathrm{s}})^{\mathrm{T}} \ (\boldsymbol{w}_n^{\mathrm{v}})^{\mathrm{T}} \right)$$

to be the driving noise vector. The augmented transition matrix is then given by

$$\boldsymbol{\Phi}_n = \begin{pmatrix} \boldsymbol{\Phi}_n^{\mathrm{s}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi}_n^{\mathrm{v}} \end{pmatrix}$$

where $\mathbf{0}$ is an all-zeros matrix of the proper dimensions. Then the concatenated recursive model is given by

$$\boldsymbol{x}_n = \boldsymbol{\Phi}_n \boldsymbol{x}_{n-1} + \boldsymbol{w}_n, \quad n = 1, 2, \ldots \tag{8.57}$$

The respective driving noise covariance matrix is given by

$$\mathbf{Q}_n = \begin{pmatrix} \mathbf{Q}_n^{\mathrm{s}} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_n^{\mathrm{v}} \end{pmatrix} .$$

Accordingly, the measurement equation can be reformulated as

$$\boldsymbol{y}_n = \mathbf{H}_n \boldsymbol{x}_n , \quad n = 1, 2, \ldots , \tag{8.58}$$

where

$$\mathbf{H}_n = \left(\mathbf{H}_n^{\mathrm{s}} \mathbf{H}_n^{\mathrm{v}}\right) .$$

Note, that as the measurement equation (8.58) is noise-free, the noise correlation matrix is zero, i. e., $\mathbf{R}_n = \mathbf{0}$.

8.3 Extensions of the Kalman Filter

8.3.1 The Kalman Predictor

In various applications it is desirable to predict the value of a future state, say $\boldsymbol{x}_m$, from measurements $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots , \boldsymbol{y}_n$ up to some time instant $n < m$. The optimal linear estimator (*predictor*) in the sense of minimum MSE is the Kalman predictor, which is based on the Kalman filter and has a simple, intuitively appealing structure.

Let us first consider a one-step predictor, which predicts (estimates) $\boldsymbol{x}_{n+1}$ from $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots , \boldsymbol{y}_n$. The prediction equation is given by

$$\hat{\boldsymbol{x}}_{n+1|n} = \boldsymbol{\Phi}_{n+1} \hat{\boldsymbol{x}}_{n|n} , \tag{8.59}$$

where $\hat{\boldsymbol{x}}_{n|n}$ is the Kalman filter's output at time instant n, namely the optimal linear estimate of $\boldsymbol{x}_n$ from the same measurements. As shown in Sect. 8.1, in order to establish optimality of the proposed predictor, we need to verify linearity, zero-mean estimation errors, and orthogonality of the estimation errors to all of the measurements $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots , \boldsymbol{y}_n$ on which the prediction is based. The linearity property is trivially satisfied from the linearity of the expression (8.59) and of the Kalman filter. To examine the other two properties, let us first establish a recursive relation for the prediction error. Subtracting $\boldsymbol{x}_{n+1}$ from both sides of (8.59), we get

$$\begin{aligned} \boldsymbol{\epsilon}_{n+1|n} &= \boldsymbol{\Phi}_{n+1} \hat{\boldsymbol{x}}_{n|n} - \boldsymbol{x}_{n+1} \\ &= \boldsymbol{\Phi}_{n+1} \hat{\boldsymbol{x}}_{n|n} - (\boldsymbol{\Phi}_{n+1} \boldsymbol{x}_n + \boldsymbol{w}_{n+1}) \\ &= \boldsymbol{\Phi}_{n+1} \boldsymbol{\epsilon}_{n|n} - \boldsymbol{w}_{n+1} , \end{aligned} \tag{8.60}$$

where we have substituted the model equation (8.17) for $\boldsymbol{x}_{n+1}$. It is now easily verified that the optimality of the Kalman filter induces optimality of the one-step predictor, as

$$E(\boldsymbol{\epsilon}_{n+1|n}) = \boldsymbol{\Phi}_{n+1} E(\boldsymbol{\epsilon}_{n|n}) - E(\boldsymbol{w}_{n+1}) = \mathbf{0} \tag{8.61}$$

and

$$E(\boldsymbol{\epsilon}_{n+1|n} \boldsymbol{y}_\ell) = \boldsymbol{\Phi}_{n+1} E(\boldsymbol{\epsilon}_{n|n} \boldsymbol{y}_\ell) - E(\boldsymbol{w}_{n+1} \boldsymbol{y}_\ell) = \mathbf{0} \tag{8.62}$$

for all $\ell = 1, 2, \ldots , n$, where we have also exploited the orthogonality of $\boldsymbol{w}_{n+1}$ to past measurements (which are linear functions of $\boldsymbol{x}_0$, past of $\boldsymbol{w}_k$ and past of $\boldsymbol{v}_k$, all of which are orthogonal to $\boldsymbol{w}_{n+1}$).

This establishes the optimality of the proposed one-step Kalman predictor. The associated MSE is also easily derived from (8.60),

$$\mathbf{P}_{n+1|n} = \boldsymbol{\Phi}_{n+1} \mathbf{P}_{n|n} \boldsymbol{\Phi}_{n+1|n}^{\mathrm{T}} + \mathbf{Q}_{n+1} , \tag{8.63}$$

where we have exploited the orthogonality of $\boldsymbol{w}_{n+1}$ to $\boldsymbol{\epsilon}_{n|n}$. In fact, the expressions obtained so far are reminiscent of the prediction steps (p1) and (p2) of the Kalman filter, in which the optimal one-step prediction $\hat{\boldsymbol{x}}_{n|n-1}$ is computed. Indeed, with one-step forward shifts of the indices we naturally get similar expressions.

Now suppose that we have the optimal linear k-step prediction and are interested in the $k+1$-step prediction. The optimal linear $k+1$-step predictor would have a structure similar to the one-step predictor, but would be based on the k-step predictor, rather than on the Kalman filter output:

$$\hat{\boldsymbol{x}}_{n+k+1|n} = \boldsymbol{\Phi}_{n+k+1} \hat{\boldsymbol{x}}_{n+k|n} . \tag{8.64}$$

Optimality can then be easily verified in the same way as for the one-step predictor, as the recursive relation of the prediction errors is essentially the same:

$$\begin{aligned} &\boldsymbol{\epsilon}_{n+k+1|n} \\ &= \boldsymbol{\Phi}_{n+k+1} \hat{\boldsymbol{x}}_{n+k|n} - \boldsymbol{x}_{n+k+1} \\ &= \boldsymbol{\Phi}_{n+k+1} \hat{\boldsymbol{x}}_{n+k|n} - (\boldsymbol{\Phi}_{n+k+1} \boldsymbol{x}_{n+k} + \boldsymbol{w}_{n+k+1}) \\ &= \boldsymbol{\Phi}_{n+k+1} \boldsymbol{\epsilon}_{n+k|n} - \boldsymbol{w}_{n+k+1} . \end{aligned} \tag{8.65}$$

The optimality propagates in exactly the same way, as orthogonality of $\boldsymbol{w}_{n+k+1}$ to past measurements and prediction errors is maintained. Likewise, the MSE can be

expressed as

$$\mathbf{P}_{n+k+1|n} = \boldsymbol{\Phi}_{n+k+1}\mathbf{P}_{n+k|n}\boldsymbol{\Phi}^{\mathrm{T}}_{n+k+1} + \mathbf{Q}_{n+k+1} \,. \tag{8.66}$$

We have therefore established, by induction, the optimality of a *propagating predictor* based on the Kalman filter output. If the intermediate predictors are not required, the general d-step predictor can be computed directly from the Kalman filter's output as

$$\hat{\boldsymbol{x}}_{m|n} = \boldsymbol{\Phi}_m\boldsymbol{\Phi}_{m-1}\cdots\boldsymbol{\Phi}_{n+1}\hat{\boldsymbol{x}}_{n|n} \,, \tag{8.67}$$

which simply propagates the model equations from time instant n to time instant m. The matrix products $\boldsymbol{\Phi}_m\boldsymbol{\Phi}_{m-1}\cdots\boldsymbol{\Phi}_{n+1}$ can be computed in advance, so that only one matrix multiplication per prediction is required in real time.

As to computation of the MSE, the recursive form of (8.66) must be maintained, beginning with $\mathbf{P}_{n+1|n}$ and ending with $\mathbf{P}_{m|n}$. Naturally, the Kalman predictor's MSE $\mathbf{P}_{m|n}$ in estimating $\boldsymbol{x}_m$ from measurements up to time instant n will always be greater than or equal to (usually greater than) the Kalman filter's error $\mathbf{P}_{m|m}$ in estimating the same state from measurements up to time instant m.

8.3.2 The Kalman Smoother

As opposed to the Kalman *predictor*, the Kalman *smoother* is aimed at estimating the state $\boldsymbol{x}_n$ from measurements $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_m$, where $m > n$, namely to base the estimate of $\boldsymbol{x}_n$ on *future* measurements, as well as on past and present measurements. The output of the optimal linear (Kalman) *smoother* must always be at least as good as (and is often much better than) the output of the optimal linear (Kalman) *filter* in terms of estimation MSE, since it optimally accounts for more measurements for estimation of $\boldsymbol{x}_n$. Naturally, such a *smoothing* scheme is not feasible in a causal real-time system. However, in view of the potential performance gain, it is sometime desirable to trade affordable latency for improved accuracy, namely to *delay* the output of the filter until some further measurements are obtained, thereby making the filter smoother. In addition, in non-real-time applications it is sometimes possible to work in an *offline* batch mode, in which it is possible to *go back in time* and improve all of the causal filter's estimates based on the entire batch of measurements.

It is common practice to distinguish between three different paradigms for smoothing, depending on the associated application: fixed-interval smoothing, fixed-point smoothing, and fixed-lag smoothing. We shall now discuss each type separately, with emphasis on the fixed-interval smoother, which can be viewed as the most general of the three, as it can also be used in the context of the other two.

Fixed-Interval Smoothing

A fixed-interval smoother is usually (but not necessarily) used in the context of batch-mode processing: once a batch of N measurements $\boldsymbol{y}_1, \boldsymbol{y}_2 \ldots, \boldsymbol{y}_N$ has been obtained, and the Kalman filter output $\hat{\boldsymbol{x}}_{n|n}$ has been computed for all $n = 1, 2, \ldots, N$, it is desired to go back and compute, for each n, a *smoothed* estimate of $\boldsymbol{x}_n$, based on the entire batch of measurements, which is therefore denoted by $\hat{\boldsymbol{x}}_{n|N}$. The fixed-interval Kalman smoother offers a backwards-recursive scheme for obtaining the desired estimates. Some fixed-interval smoothers (see, e.g., [8.8]) employ three-pass smoothing: in the first pass, the ordinary Kalman filter is applied; in the second pass, a backwards Kalman filter is applied; in the third pass the results of both passes are optimally combined to attain the smoothed output.

However, herein we shall review the Rauch–Tung–Striebel two-pass smoother (derived by *H. Rauch*, *K. Tung* and *C. Striebel* in 1965 [8.9]). The two-pass smoother employs just one (the second) pass on the data in addition to the original (Kalman filter) pass. This second pass begins with the estimate of the last state $\hat{\boldsymbol{x}}_{N|N}$ (which is both the filtered and the smoothed estimate for $n = N$, since N is the last sample in the batch), and propagates backwards as $n = N-1, N-2, \ldots, 1$, generating the smoothed estimate $\hat{\boldsymbol{x}}_{n|N}$ at each time instant n from the previously obtained Kalman filter's outputs $\hat{\boldsymbol{x}}_{n|n}$ and $\hat{\boldsymbol{x}}_{n+1|n}$ and from the smoothed output of the succeeding time instant, $\hat{\boldsymbol{x}}_{n+1|N}$.

For simplicity of notation, we shall denote the Kalman smoother's output (at time instant n) as $\hat{\boldsymbol{x}}_n$, rather than $\hat{\boldsymbol{x}}_{n|N}$. The form $\hat{\boldsymbol{x}}_{n|m}$ will still be used to denote the Kalman filter's output. As already mentioned, for $n = N$ we have $\hat{\boldsymbol{x}}_N = \hat{\boldsymbol{x}}_{N|N}$, namely the smoother's output equals the Kalman filter's output, which is already the optimal linear estimate of $\boldsymbol{x}_N$ based on the entire batch. For $n = N-1, N-2, \ldots, 1$, the following backwards-recursive scheme is proposed:

$$\hat{\boldsymbol{x}}_n = \hat{\boldsymbol{x}}_{n|n} + \mathbf{A}_n(\hat{\boldsymbol{x}}_{n+1} - \hat{\boldsymbol{x}}_{n+1|n}) \,, \quad n = N-1, N-2, \ldots, 1 \,, \tag{8.68}$$

where $\mathbf{A}_n$ is a gain matrix given by

$$\mathbf{A}_n = \mathbf{P}_{n|n}\boldsymbol{\Phi}^{\mathrm{T}}_{n+1}\mathbf{P}^{-1}_{n+1|n} \,. \tag{8.69}$$

At each time instant n, the smoother *corrects* the filter by adding a term that is proportional (via $\mathbf{A}_n$) to the difference between the smoother's output and the filter's predicted output for $n+1$. We shall now show the optimality of this scheme by verifying the three optimality conditions outlined in Sect. 8.1, namely: linearity, zero mean of the estimation error, and orthogonality of the estimation errors to the measurements $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N$ on which the estimates are based.

The linearity condition is trivially satisfied by the proposed structure, dwelling on the linearity of the Kalman filter. To verify the other two conditions, we first obtain recursion relations for the estimation error, denoted by $\boldsymbol{\epsilon}_n = \hat{\boldsymbol{x}}_n - \boldsymbol{x}_n$. By subtracting $\boldsymbol{x}_n$ from both sides of (8.68) and $\boldsymbol{x}_{n+1}$ from both terms in the parentheses we obtain

$$\boldsymbol{\epsilon}_n = \boldsymbol{\epsilon}_{n|n} + \mathbf{A}_n(\boldsymbol{\epsilon}_{n+1} - \boldsymbol{\epsilon}_{n+1|n}) \,, \tag{8.70}$$

where $\boldsymbol{\epsilon}_{n|n}$ and $\boldsymbol{\epsilon}_{n+1|n}$ are the Kalman filter errors, defined in Sect. 8.1. Due to the optimality of the Kalman filter these errors have zero mean and are orthogonal to all $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n$.

Exploiting the zero-mean property in (8.70), we obtain $E(\boldsymbol{\epsilon}_n) = \mathbf{A}_n E(\boldsymbol{\epsilon}_{n+1})$. Recalling that $\boldsymbol{\epsilon}_N = \boldsymbol{\epsilon}_{N|N}$ (whose mean is zero), the zero-mean property is thereby induced backwards for all n.

We now consider the orthogonality property. From (8.70) we obtain

$$\begin{aligned} E(\boldsymbol{\epsilon}_n \boldsymbol{y}_\ell^{\mathrm{T}}) &= E(\boldsymbol{\epsilon}_{n|n} \boldsymbol{y}_\ell^{\mathrm{T}}) + \mathbf{A}_n E(\boldsymbol{\epsilon}_{n+1} \boldsymbol{y}_\ell^{\mathrm{T}}) \\ &\quad - \mathbf{A}_n E(\boldsymbol{\epsilon}_{n+1|n} \boldsymbol{y}_\ell^{\mathrm{T}}) \;\; \forall \ell \,. \end{aligned} \tag{8.71}$$

Due to the optimality of the Kalman filter and to the assumed optimality of $\hat{\boldsymbol{x}}_{n+1}$, we immediately obtain $E(\boldsymbol{\epsilon}_n \boldsymbol{y}_\ell^{\mathrm{T}}) = \mathbf{0}$ for all $\ell \le n$. Obviously, to establish the smoother's optimality it now remains to show the orthogonality of $\boldsymbol{\epsilon}_n$ to $\boldsymbol{y}_\ell$ for $n < \ell \le N$. Let us therefore consider the case $\ell = n+k$ for positive values of k. Based on the assumed optimality of $\hat{\boldsymbol{x}}_{n+1}$, we obtain

$$\begin{aligned} &E\big(\boldsymbol{\epsilon}_n \boldsymbol{y}_{n+k}^{\mathrm{T}}\big) \\ &= E\big(\boldsymbol{\epsilon}_{n|n} \boldsymbol{y}_{n+k}^{\mathrm{T}}\big) - \mathbf{A}_n E\big(\boldsymbol{\epsilon}_{n+1|n} \boldsymbol{y}_{n+k}^{\mathrm{T}}\big) \\ &= E\big[\boldsymbol{\epsilon}_{n|n} (\mathbf{H}_{n+k} \boldsymbol{x}_{n+k} + \boldsymbol{v}_{n+k})^{\mathrm{T}}\big] \\ &\quad - \mathbf{A}_n E\big[\boldsymbol{\epsilon}_{n+1|n} (\mathbf{H}_{n+k} \boldsymbol{x}_{n+k} + \boldsymbol{v}_{n+k})^{\mathrm{T}}\big] \\ &= E\big(\boldsymbol{\epsilon}_{n|n} \boldsymbol{x}_{n+k}^{\mathrm{T}}\big) \mathbf{H}_{n+k}^{\mathrm{T}} - \mathbf{A}_n E\big(\boldsymbol{\epsilon}_{n+1|n} \boldsymbol{x}_{n+k}^{\mathrm{T}}\big) \mathbf{H}_{n+k}^{\mathrm{T}} \\ &= \big[E\big(\boldsymbol{\epsilon}_{n|n} \boldsymbol{x}_{n+k}^{\mathrm{T}}\big) - \mathbf{A}_n E\big(\boldsymbol{\epsilon}_{n+1|n} \boldsymbol{x}_{n+k}^{\mathrm{T}}\big)\big] \mathbf{H}_{n+k}^{\mathrm{T}} \,, \end{aligned} \tag{8.72}$$

where the third equality is due to orthogonality of $\boldsymbol{\epsilon}_{n|n}$ and $\boldsymbol{\epsilon}_{n+1|n}$ to future measurement noise $\boldsymbol{v}_{n+k}$ for all positive k. We shall now show that the term in parentheses in (8.72) is zero. Consider first the case $k = 1$. The term in parentheses is then given by

$$\begin{aligned} &E\big(\boldsymbol{\epsilon}_{n|n} \boldsymbol{x}_{n+1}^{\mathrm{T}}\big) - \mathbf{A}_n E\big(\boldsymbol{\epsilon}_{n+1|n} \boldsymbol{x}_{n+1}^{\mathrm{T}}\big) \\ &= E\big(\boldsymbol{\epsilon}_{n|n} (\boldsymbol{\Phi}_{n+1} \boldsymbol{x}_n + \boldsymbol{w}_{n+1})^{\mathrm{T}}\big) \\ &\quad - \mathbf{A}_n E\big[\boldsymbol{\epsilon}_{n+1|n} (\hat{\boldsymbol{x}}_{n+1|n} - \boldsymbol{\epsilon}_{n+1|n})^{\mathrm{T}}\big] \\ &= E\big(\boldsymbol{\epsilon}_{n|n} \boldsymbol{x}_n^{\mathrm{T}}\big) \boldsymbol{\Phi}_{n+1}^{\mathrm{T}} + \mathbf{A}_n E\big(\boldsymbol{\epsilon}_{n+1|n} \boldsymbol{\epsilon}_{n+1|n}^{\mathrm{T}}\big) \\ &= E\big[\boldsymbol{\epsilon}_{n|n} (\hat{\boldsymbol{x}}_{n|n} - \boldsymbol{\epsilon}_{n|n})^{\mathrm{T}}\big] \boldsymbol{\Phi}_{n+1}^{\mathrm{T}} + \mathbf{A}_n \mathbf{P}_{n+1|n} \\ &= -\mathbf{P}_{n|n} \boldsymbol{\Phi}_{n+1}^{\mathrm{T}} + \mathbf{A}_n \mathbf{P}_{n+1|n} = \mathbf{0} \,, \end{aligned} \tag{8.73}$$

where the transitions rely on orthogonality of $\boldsymbol{\epsilon}_{n|n}$ to $\boldsymbol{w}_{n+1}$, as well as on the orthogonality of both $\boldsymbol{\epsilon}_{n|n}$ and $\boldsymbol{\epsilon}_{n+1|n}$ to the Kalman filter's outputs $\hat{\boldsymbol{x}}_{n|n}$ and $\hat{\boldsymbol{x}}_{n+1|n}$, which are linear functions of measurements to which these errors are orthogonal. The last equality is obtained by substituting $\mathbf{A}_n$ from (8.69).

Now consider, by induction, the case of $k > 1$. Assume that the expression in parentheses in (8.72) is zero for some k, and consider $k+1$, for which the expression can be written as

$$\begin{aligned} &E\big(\boldsymbol{\epsilon}_{n|n} \boldsymbol{x}_{n+k+1}^{\mathrm{T}}\big) - \mathbf{A}_n E\big(\boldsymbol{\epsilon}_{n+1|n} \boldsymbol{x}_{n+k+1}^{\mathrm{T}}\big) \\ &= E\big[\boldsymbol{\epsilon}_{n|n} (\boldsymbol{\Phi}_{n+k+1} \boldsymbol{x}_{n+k} + \boldsymbol{w}_{n+k+1})^{\mathrm{T}}\big] \\ &\quad - \mathbf{A}_n E\big[\boldsymbol{\epsilon}_{n+1|n} (\boldsymbol{\Phi}_{n+k+1} \boldsymbol{x}_{n+k} + \boldsymbol{w}_{n+k+1})^{\mathrm{T}}\big] \\ &= E\big(\boldsymbol{\epsilon}_{n|n} \boldsymbol{x}_{n+k}^{\mathrm{T}}\big) \boldsymbol{\Phi}_{n+k+1}^{\mathrm{T}} \\ &\quad - \mathbf{A}_n E\big(\boldsymbol{\epsilon}_{n+1|n} \boldsymbol{x}_{n+k}^{\mathrm{T}}\big) \boldsymbol{\Phi}_{n+k+1}^{\mathrm{T}} \\ &= \big[E\big(\boldsymbol{\epsilon}_{n|n} \boldsymbol{x}_{n+k}^{\mathrm{T}}\big) - \mathbf{A}_n E\big(\boldsymbol{\epsilon}_{n+1|n} \boldsymbol{x}_{n+k}^{\mathrm{T}}\big)\big] \boldsymbol{\Phi}_{n+k+1}^{\mathrm{T}} \\ &= \mathbf{0} \,, \end{aligned} \tag{8.74}$$

where the second equality is due to the orthogonality of $\boldsymbol{\epsilon}_{n|n}$ and $\boldsymbol{\epsilon}_{n+1|n}$ to $\boldsymbol{w}_{n+k+1}$. This established the optimality of the proposed smoother. A remaining important issue is the resulting MSE, expressed by $\mathbf{P}_n = E(\boldsymbol{\epsilon}_n \boldsymbol{\epsilon}_n^{\mathrm{T}})$ (we maintain the notation simplification in using $\mathbf{P}_n$ instead of $\mathbf{P}_{n|N}$). Note that, while in the Kalman filter the computation of $\mathbf{P}_{n|n}$ as well as $\mathbf{P}_{n|n-1}$ is inherent to the filtering equation and is therefore obtained as a byproduct, the smoother does not require computation of $\mathbf{P}_n$. Nevertheless, we shall now derive a recursive expression for this important measure of performance.

Obviously, we have $\mathbf{P}_N = \mathbf{P}_{N|N}$. Let us now rewrite (8.70) as:

$$\boldsymbol{\epsilon}_n - \mathbf{A}_n \boldsymbol{\epsilon}_{n+1} = \boldsymbol{\epsilon}_{n|n} - \mathbf{A}_n \boldsymbol{\epsilon}_{n+1|n} \,. \tag{8.75}$$

Multiplying each side by its transpose and taking the mean, we obtain

$$\begin{aligned}&\mathbf{P}_n+\mathbf{A}_n\mathbf{P}_{n+1}\mathbf{A}_n^{\mathrm{T}}-E\big(\boldsymbol{\epsilon}_n\boldsymbol{\epsilon}_{n+1}^{\mathrm{T}}\big)\mathbf{A}_n^{\mathrm{T}}-\mathbf{A}_nE\big(\boldsymbol{\epsilon}_{n+1}\boldsymbol{\epsilon}_n^{\mathrm{T}}\big)\\&=\mathbf{P}_{n|n}+\mathbf{A}_n\mathbf{P}_{n+1|n}\mathbf{A}_n^{\mathrm{T}}\\&\quad-E\big(\boldsymbol{\epsilon}_{n|n}\boldsymbol{\epsilon}_{n+1|n}^{\mathrm{T}}\big)\mathbf{A}_n^{\mathrm{T}}-\mathbf{A}_nE\big(\boldsymbol{\epsilon}_{n+1|n}\boldsymbol{\epsilon}_{n|n}^{\mathrm{T}}\big)\,.\end{aligned}\tag{8.76}$$

In order to obtain a closed-form recursive expression, we need to evaluate the terms $E(\boldsymbol{\epsilon}_n\boldsymbol{\epsilon}_{n+1}^{\mathrm{T}})$ and $E(\boldsymbol{\epsilon}_{n|n}\boldsymbol{\epsilon}_{n+1|n}^{\mathrm{T}})$. Regarding the latter, we observe that [using the recursion in (8.22)]

$$\begin{aligned}E\big(\boldsymbol{\epsilon}_{n|n}\boldsymbol{\epsilon}_{n+1|n}^{\mathrm{T}}\big)&=E\big[\boldsymbol{\epsilon}_{n|n}(\boldsymbol{\Phi}_{n+1}\boldsymbol{\epsilon}_{n|n}-\boldsymbol{w}_{n+1})^{\mathrm{T}}\big]\\&=\mathbf{P}_{n|n}\boldsymbol{\Phi}_{n+1}^{\mathrm{T}}=\mathbf{A}_n\mathbf{P}_{n+1|n}\,,\end{aligned}\tag{8.77}$$

where we have exploited the orthogonality of $\boldsymbol{\epsilon}_{n|n}$ to $\boldsymbol{w}_{n+1}$, as well as the expression for $\mathbf{A}_n$ (8.69). Regarding the other term, let us first prove the identity

$$E\big[(\boldsymbol{\epsilon}_n-\mathbf{A}_n\boldsymbol{\epsilon}_{n+1})\boldsymbol{\epsilon}_{n+1}^{\mathrm{T}}\big]=\mathbf{0}\,.\tag{8.78}$$

To show this, note that

$$\begin{aligned}&E\big[(\boldsymbol{\epsilon}_n-\mathbf{A}_n\boldsymbol{\epsilon}_{n+1})\boldsymbol{\epsilon}_{n+1}^{\mathrm{T}}\big]\\&=E\big[(\boldsymbol{\epsilon}_n-\mathbf{A}_n\boldsymbol{\epsilon}_{n+1})(\hat{\boldsymbol{x}}_{n+1}-\boldsymbol{x}_{n+1})^{\mathrm{T}}\big]\\&=-E\big[(\boldsymbol{\epsilon}_n-\mathbf{A}_n\boldsymbol{\epsilon}_{n+1})\boldsymbol{x}_{n+1}^{\mathrm{T}}\big]\\&=-E\big[(\boldsymbol{\epsilon}_{n|n}-\mathbf{A}_n\boldsymbol{\epsilon}_{n+1|n})\boldsymbol{x}_{n+1}^{\mathrm{T}}\big]\,,\end{aligned}\tag{8.79}$$

where we have used the orthogonality of $\boldsymbol{\epsilon}_n$ and $\boldsymbol{\epsilon}_{n+1}$ to $\hat{\boldsymbol{x}}_{n+1}$ (a linear function of all the measurements, to which both $\boldsymbol{\epsilon}_n$ and $\boldsymbol{\epsilon}_{n+1}$ are orthogonal), as well as the relation (8.75). Now,

$$\begin{aligned}&E\big(\boldsymbol{\epsilon}_{n|n}\boldsymbol{x}_{n+1}^{\mathrm{T}}\big)=E\big[\boldsymbol{\epsilon}_{n|n}(\boldsymbol{\Phi}_{n+1}\boldsymbol{x}_n+\boldsymbol{w}_{n+1})^{\mathrm{T}}\big]\\&=E\big(\boldsymbol{\epsilon}_{n|n}\boldsymbol{x}_n^{\mathrm{T}}\big)\boldsymbol{\Phi}_{n+1}^{\mathrm{T}}=E\big[\boldsymbol{\epsilon}_{n|n}(\hat{\boldsymbol{x}}_{n|n}-\boldsymbol{\epsilon}_{n|n})^{\mathrm{T}}\big]\boldsymbol{\Phi}_{n+1}^{\mathrm{T}}\\&=-\mathbf{P}_{n|n}\boldsymbol{\Phi}_{n+1}^{\mathrm{T}}\end{aligned}\tag{8.80}$$

(again, due to orthogonality of $\boldsymbol{\epsilon}_{n|n}$ to $\boldsymbol{w}_{n+1}$ and to $\hat{\boldsymbol{x}}_{n|n}$), and

$$\begin{aligned}&E\big(\mathbf{A}_n\boldsymbol{\epsilon}_{n+1|n}\boldsymbol{x}_{n+1}^{\mathrm{T}}\big)\\&=\mathbf{A}_nE\big[\boldsymbol{\epsilon}_{n+1|n}(\hat{\boldsymbol{x}}_{n+1|n}-\boldsymbol{\epsilon}_{n+1|n})^{\mathrm{T}}\big]\\&=-\mathbf{A}_n\mathbf{P}_{n+1|n}=-\mathbf{P}_{n|n}\boldsymbol{\Phi}_{n+1}^{\mathrm{T}}\end{aligned}\tag{8.81}$$

[due to the orthogonality of $\boldsymbol{\epsilon}_{n+1|n}$ to $\hat{\boldsymbol{x}}_{n+1|n}$ and due to (8.69)]. Substituting (8.80) and (8.81) into (8.79), we obtain the desired identity (8.78). Now, the expression on the left-hand side of (8.78) can also be written as

$$\begin{aligned}&E\big[(\boldsymbol{\epsilon}_n-\mathbf{A}_n\boldsymbol{\epsilon}_{n+1})\boldsymbol{\epsilon}_{n+1}^{\mathrm{T}}\big]\\&=E\big(\boldsymbol{\epsilon}_n\boldsymbol{\epsilon}_{n+1}^{\mathrm{T}}\big)-\mathbf{A}_nE\big[\boldsymbol{\epsilon}_{n+1}\boldsymbol{\epsilon}_{n+1}^{\mathrm{T}}\big]\,.\end{aligned}\tag{8.82}$$

Since this term has just been shown to vanish, we deduce that

$$E\big(\boldsymbol{\epsilon}_n\boldsymbol{\epsilon}_{n+1}^{\mathrm{T}}\big)=\mathbf{A}_n\mathbf{P}_{n+1}\,.\tag{8.83}$$

We can now substitute the missing expressions (8.77) and (8.83) (and their transposed forms) into (8.76), obtaining

$$\begin{aligned}&\mathbf{P}_n+\mathbf{A}_n\mathbf{P}_{n+1}\mathbf{A}_n^{\mathrm{T}}-2\mathbf{A}_n\mathbf{P}_{n+1}\mathbf{A}_n^{\mathrm{T}}\\&=\mathbf{P}_{n|n}+\mathbf{A}_n\mathbf{P}_{n+1|n}\mathbf{A}_n^{\mathrm{T}}-2\mathbf{A}_n\mathbf{P}_{n+1|n}\mathbf{A}_n^{\mathrm{T}}\,.\end{aligned}\tag{8.84}$$

We thus obtain the following (backwards-)recursive expression for $\mathbf{P}_n$:

$$\begin{aligned}\mathbf{P}_n&=\mathbf{P}_{n|n}-\mathbf{A}_n(\mathbf{P}_{n+1|n}-\mathbf{P}_{n+1})\mathbf{A}_n^{\mathrm{T}}\,,\\&\qquad n=N-1,N-2,\ldots,1\,,\end{aligned}\tag{8.85}$$

in which it is nicely observed that, as expected, for each n the smoother's MSE $\mathbf{P}_n$ is smaller than or equal to (usually smaller than) the filter's MSE $\mathbf{P}_{n|n}$. This is because the smoothed estimate is optimally based on more measurements than the filtered estimate. To observe this relation, note that for each $n<N$ we have $\mathbf{P}_{n+1|n}\geq\mathbf{P}_{n+1|n+1}$, and, since for $n=N$ we have $\mathbf{P}_{N|N}=\mathbf{P}_N$, we get, by backwards induction, $\mathbf{P}_{n+1|n}\geq\mathbf{P}_{n+1|n+1}\geq\mathbf{P}_{n+1}$ for all $n<N$.

Fixed-Lag Smoothing

A fixed-lag smoother is aimed at estimating the state at time $n-d$ from measurements up to time n, where d is some fixed positive integer (the *lag*). As n progresses, the smoother generates the estimates $\hat{\boldsymbol{x}}_{n-d|n}$, trading a delay of d measurements (in real-time contexts) for improved accuracy in the estimation of $\boldsymbol{x}_{n-d}$. We shall not specify the specific smoothing equations in here; note only that the fixed-interval smoother derived above can also be used for fixed-lag smoothing by applying d backwards-recursions at each time instant. However, such a scheme is computationally more expensive than applying a true fixed-lag smoother, especially for large values of d.

It is important to note, that in some applications (mainly signal denoising) the state vector $\boldsymbol{x}$ contains past-samples of the underlying signal — see, e.g., the case of AR sources in Sect. 8.2. In such cases, estimates of some fixed-lag past samples of *the underlying signal(s)* (as opposed to past samples of *the entire state*) are readily available from the Kalman filter's state estimate, eliminating the need for a smoother. The maximum available lag is determined by the oldest sample contained in $\boldsymbol{x}_n$. For example, in the case of estimating AR signals in white noise, it is the AR order minus 1. If desired, it is theoretically possible to increase the size of the state vector artificially so as to include more past samples that would automatically be estimated at the filter's

output, thus eliminating the need for explicit smoothing, but at the cost of increased dimensionality of the Kalman filter. However, usually this trade-off has more conceptual than computational appeal.

Fixed-Point Smoothing

A fixed-point smoother is aimed at estimating the state at a fixed time instant m from measurements up to time n, extending beyond m. Namely, as n progresses, the smoother keeps updating its estimate $\hat{\boldsymbol{x}}_{m|n}$. Again, we shall not include the explicit fixed-point smoothing equation in here; note only that, in principle, the fixed-interval smoother can again be used to compute the fixed-point smoothing, but at significantly higher computational cost, especially as n departs farther from m.

8.3.3 The Extended Kalman Filter

In many scenarios the assumption of linear transition and/or measurement models is not realistic. In fact, the first full implementation that transformed the Kalman filter into a useful working tool was done in the Ames Research Center of the NASA in Mountain View, California in 1960 [8.10], where the trajectory estimation for the Apollo project (the first manned mission to the moon) was studied at the time. In that context, the linear model was observed to be inadequate for describing the trajectory of interest. The resulting extended Kalman filter (EKF) was derived by Stanley F. Schmidt after Kalman's visit. This extension of the Kalman filter is still the most popular tool for dealing with nonlinear applications. However, it should be stressed that, as opposed to the case of a linear model, no claims of optimality hold, and that the EKF is merely a heuristic extension of the linear Kalman filter.

Consider the nonlinear recursive system:

$$\boldsymbol{x}_n = \boldsymbol{\phi}_n(\boldsymbol{x}_{n-1}) + \boldsymbol{w}_n, \quad n = 1, 2, \ldots , \tag{8.86}$$

where $\boldsymbol{\phi}_n(\cdot)$ is a nonlinear vector function of a vector input. $\boldsymbol{w}_n$ is the temporally uncorrelated driving noise. Consider also the nonlinear measurement equation

$$\boldsymbol{y}_n = \boldsymbol{h}_n(\boldsymbol{x}_n) + \boldsymbol{v}_n, \quad n = 1, 2, \ldots , \tag{8.87}$$

where $\boldsymbol{h}_n(\cdot)$ is a nonlinear vector function of a vector input. $\boldsymbol{v}_n$ is the temporally uncorrelated measurement noise.

The simple idea behind the EKF is to approximate the nonlinear functions by their first-order Taylor series expansion about the estimated trajectory [8.4, 5]. Let,

$$\boldsymbol{\Phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1}) = \frac{\partial \boldsymbol{\phi}_n(\boldsymbol{x})}{\partial \boldsymbol{x}}|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{n-1|n-1}}$$

be the derivative matrix of the nonlinear transition function with respect to its argument at the current estimate, namely, the (k, ℓ)-th element of $\boldsymbol{\Phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1})$ is the derivative of the k-th element of $\boldsymbol{\phi}_n(\boldsymbol{x})$ with respect to the ℓ-th element of $\boldsymbol{x}$, evaluated at $\boldsymbol{x} = \hat{\boldsymbol{x}}_{n-1|n-1}$. Then, using a first-order Taylor approximation:

$$\begin{aligned}\boldsymbol{x}_n \approx\ & \boldsymbol{\phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1}) + \boldsymbol{\Phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1}) \\ & \cdot (\boldsymbol{x}_n - \hat{\boldsymbol{x}}_{n-1|n-1}) + \boldsymbol{w}_n, \quad n = 1, 2, \ldots\end{aligned} \tag{8.88}$$

In the *propagation* step, based on the rationale of the linear Kalman filter, we compute the estimate of $\boldsymbol{x}_n$ by applying the model equation to $\hat{\boldsymbol{x}}_{n-1|n-1}$, ignoring the unknown, zero-mean driving noise $\boldsymbol{w}_n$.

$$\hat{\boldsymbol{x}}_{n|n-1} = \boldsymbol{\phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1}) \, . \tag{8.89}$$

To calculate the corresponding covariance we first derive the error term. Let,

$$\begin{aligned}\boldsymbol{\epsilon}_{n|n-1} &= \hat{\boldsymbol{x}}_{n|n-1} - \boldsymbol{x}_n = \boldsymbol{\phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1}) - \boldsymbol{x}_n \\ &\approx \boldsymbol{\phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1}) - (\boldsymbol{\phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1}) \\ &\quad + \boldsymbol{\Phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1})(\boldsymbol{x}_n - \hat{\boldsymbol{x}}_{n-1|n-1}) + \boldsymbol{w}_n) \\ &= \boldsymbol{\Phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1})\boldsymbol{\epsilon}_{n-1|n-1} - \boldsymbol{w}_n \, .\end{aligned} \tag{8.90}$$

In the context of nonlinear models, it is common practice to assume that the driving noise $\boldsymbol{w}_n$ and the measurement noise $\boldsymbol{v}_n$ are not only uncorrelated sequences, but are also statistically independent sequences and are also statistically independent of the initial state $\boldsymbol{x}_0$. Using this assumption, the estimation error $\boldsymbol{\epsilon}_{n-1|n-1}$ can be assumed statistically independent of the *future* driving noise $\boldsymbol{w}_n$, hence $E(\boldsymbol{\epsilon}_{n-1|n-1}\boldsymbol{w}_n^{\mathrm{T}}) = \mathbf{0}$. Therefore, the covariance is approximately given, under the small-estimation-error assumption, as:

$$\begin{aligned}&\mathbf{P}_{n|n-1} \\ &\approx \boldsymbol{\Phi}_n(\hat{\boldsymbol{x}}_{n-1|n-1})\mathbf{P}_{n-1|n-1}\boldsymbol{\Phi}_n^{\mathrm{T}}(\hat{\boldsymbol{x}}_{n-1|n-1}) + \mathbf{Q}_n \, .\end{aligned} \tag{8.91}$$

Now, let

$$\mathbf{H}_n(\hat{\boldsymbol{x}}_{n|n-1}) = \frac{\partial \boldsymbol{h}_n(\boldsymbol{x})}{\partial \boldsymbol{x}}|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{n|n-1}}$$

be the derivative of the nonlinear measurement function with respect to its argument at the current estimate. Then,

$$\begin{aligned}\boldsymbol{y}_n \approx\ & \boldsymbol{h}_n(\hat{\boldsymbol{x}}_{n|n-1}) + \mathbf{H}_n(\hat{\boldsymbol{x}}_{n|n-1}) \\ & \cdot (\boldsymbol{x}_n - \hat{\boldsymbol{x}}_{n|n-1}) + \boldsymbol{v}_n, \quad n = 1, 2, \ldots\end{aligned} \tag{8.92}$$

This equation can be reformulated as

$$\begin{aligned}\tilde{\boldsymbol{y}}_n &= \boldsymbol{y}_n - (\boldsymbol{h}_n(\hat{\boldsymbol{x}}_{n|n-1}) - \mathbf{H}_n(\hat{\boldsymbol{x}}_{n|n-1})\hat{\boldsymbol{x}}_{n|n-1}) \\ &\approx \mathbf{H}_n(\hat{\boldsymbol{x}}_{n|n-1})\boldsymbol{x}_n + \boldsymbol{v}_n, \quad n = 1, 2, \ldots\end{aligned} \tag{8.93}$$

Note, that $\tilde{\boldsymbol{y}}_n$ depends both on the available estimate $\hat{\boldsymbol{x}}_{n|n-1}$ and on $\boldsymbol{y}_n$. Using (8.21) the update equation becomes,

$$\begin{aligned}\hat{\boldsymbol{x}}_{n|n} &= \hat{\boldsymbol{x}}_{n|n-1} + \mathbf{K}_n(\tilde{\boldsymbol{y}}_n - \mathbf{H}_n(\hat{\boldsymbol{x}}_{n|n-1})\hat{\boldsymbol{x}}_{n|n-1}) \\ &= \hat{\boldsymbol{x}}_{n|n-1} + \mathbf{K}_n(\boldsymbol{y}_n - \boldsymbol{h}_n(\hat{\boldsymbol{x}}_{n|n-1}))\,,\end{aligned} \tag{8.94}$$

where the last transition is due to (8.93). The respective covariance is approximated by

$$\mathbf{P}_{n|n} \approx [\mathbf{I} - \mathbf{K}_n(\hat{\boldsymbol{x}}_{n|n-1})\mathbf{H}_n(\hat{\boldsymbol{x}}_{n|n-1})]\mathbf{P}_{n|n-1}\,. \tag{8.95}$$

The Kalman gain is similar to the linear case, using the linearized equations.

$$\begin{aligned}\mathbf{K}_n = &\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}}(\hat{\boldsymbol{x}}_{n|n-1}) \\ &[\mathbf{H}_n(\hat{\boldsymbol{x}}_{n|n-1})\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}}(\hat{\boldsymbol{x}}_{n|n-1}) + \mathbf{R}_n]^{-1}\,,\end{aligned} \tag{8.96}$$

Note that, as opposed to the case of the linear Kalman filter, the Kalman gain and the associated covariances cannot be calculated in advance (offline), since the matrices $\boldsymbol{\Phi}$ and $\mathbf{H}$ are, in general, data dependent.

8.4 The Application of the Kalman Filter to Speech Processing

8.4.1 Literature Survey

Speech quality and intelligibility might significantly deteriorate in the presence of background noise, especially when the speech signal is subject to subsequent processing. In particular, speech coders, and automatic speech recognition (ASR) systems, which were designed or trained to act on clean speech signals, might be rendered useless in the presence of background noise. Speech enhancement algorithms have therefore attracted a great deal of interest over the past three decades.

Among these speech enhancement algorithms there are numerous algorithms based on *Wiener* [8.11] or Kalman filtering [8.2], both of which require knowledge of the parameters involved. For example, for the application of the Kalman filter, it is common to model the speech as a quasistationary AR process [in the speech context, the AR parameters are usually called linear predictive coefficients (LPC)], which requires knowledge of the AR parameters. The noise level (and structure) should also be available for the Kalman formulation. Since these parameters are usually unknown, the problem of joint estimation of signal and parameters arises. A typical procedure, applied to this type of problems, is the estimate–maximize (EM) algorithm [8.12]. Under the associated model assumptions, this procedure is guaranteed to converge to the maximum-likelihood (ML) estimate of the problem's parameters, or at least to a local maximum of the likelihood function, and to yield an estimate of the speech signal, computed at the parameter that maximizes the likelihood.

Paliwal and *Basu* [8.13] were, perhaps, the first to use the Kalman filter in the context of speech enhancement. Their experimental results reveal its distinct advantage over the Wiener filter, due to the ability of the Kalman filter to account for the quasistationarity of the speech signal. However, in their experiment the unknown speech parameters were not estimated from the noisy speech signal, but rather from the clean signal, which is obviously unavailable in most practical situations. *Gibson* et al. [8.14, 15] proposed to extend the use of the Kalman filter by incorporating a colored-noise model for improving the enhancement performance for a certain class of noise sources. The proposed algorithm iterates between Kalman filtering of the given corrupted speech measurements, and estimation of the speech parameters given the enhanced speech waveform. Since the authors suggest using the ordinary Yule–Walker equations [8.16] for estimating the speech AR parameters, the resulting algorithm is only an approximated version of the EM algorithm. The estimated speech LPC parameters were shown to improve speech coding systems that rely on AR modeling of the speech signal.

A comprehensive study of the use of the EM algorithm in diverse problems of joint estimation of signals and parameters is given in a series of works by Weinstein et al. In [8.17], the noise cancelation problem presented by *Widrow* [8.18] is addressed using the EM algorithm in the frequency domain. The more-general two-channel noise cancelation problem is addressed in [8.19]. Both are comprised of iterations between parameter estimation and Wiener filtering. In [8.20], a time-domain formulation of the single-microphone speech enhancement problem is presented. The approach consists of representing the signal model using linear dynamic state equations, and applying the EM method. The resulting algorithm is similar in structure to the *Lim* and *Oppenheim* [8.21] algorithm, except that the noncausal

Wiener filter is replaced by the Kalman smoothing equations. Sequential speech enhancement algorithms are presented as well. These sequential algorithms are characterized by a forward Kalman filter whose parameters are continuously updated by gradient-descent search on the likelihood function. In [8.22, 23] sequential approximations to the EM algorithm are elaborated on in the context of two-channels noise cancelation. The related problem of single-microphone active noise cancelation (ANC) is presented in [8.24].

Lee et al. [8.25, 26] extend the sequential single sensor algorithm of Weinstein et al. by considering an alternative speech-generation model, in which the white Gaussian excitation sequence is replaced with a Gaussian-mixture distributed sequence, that may account for the presence of an impulse train in the excitation sequence of voiced speech. A recursive gradient-based approach is applied to the parameter estimation. Lee et al. examined the signal-to-noise ratio (SNR) improvement of the algorithm when applied to synthetic speech input. *Goh* et al. [8.27] propose different modeling of the speech excitation sequence. The proposed model is comprised of both Gaussian white noise (modeling the unvoiced part) and an impulse train (presenting the voiced part). The latter is modeled as a long-term AR process. The resulting high-dimensional Kalman filter is efficiently implemented by exploiting the sparsity of the associated matrices. The parameter estimation is conducted via EM iterations. When the standard Kalman filter gain is recursively computed, one needs to estimate the speech and noise gains. To avoid this estimation stage, *Gabrea* et al. [8.28] propose checking the whiteness of the innovation sequence to test whether the asymptotically optimal solution has been reached. Since the estimation of the AR parameters cannot be avoided, Gabrea et al. propose to use the modified Yule–Walker procedure [8.29]. Another extension to the work of Weinstein et al. was proposed by *Gannot* et al. [8.30]. In this work both iterative batch and sequential versions of the EM algorithm are considered. The estimate stage (E-step) is implemented by applying the Kalman filter. Higher-order statistics (HOS) are employed to obtain a robust initialization to the parameter estimation stage (called the maximization stage – M-step). *Fujimoto* and *Ariki* [8.31] use Kalman filtering in the frequency domain (without using the AR model). Initialization of their algorithm is obtained by the classical spectral subtraction [8.32] algorithm (see also the relevant Chap. 44 in this Handbook).

Several nonlinear extensions to the standard Kalman filter exist. *Lee* et al. [8.33] propose the application of a robust Kalman filter. Similar to other contributions, iterations between signal enhancement and parameter estimation are conducted. The novelty of that paper stems from the use of nonlinear estimation procedures. Both the parameters and signals are estimated in a robust manner by introducing a saturation function into the cost function, rather than using the standard squared cost function. *Ma* et al. [8.34] introduce the application of a postfilter based on masking properties of the human auditory system to further enhance the resulting auditory quality of the speech signal. However, not much attention is paid in this work to the estimation of the associated AR parameters.

Shen and *Deng* [8.35] present a new and interesting approach to speech enhancement based on H_∞ filtering. This approach differs from the traditional Kalman filtering approach in the definition of the error criterion for the filter design. Rather than minimizing a squared error term, as in the standard Kalman filter, their procedure consists of calculating a filter that minimizes the worst possible amplification of the estimation error in terms of the modeling errors and additive noises. The parameters are estimated in parallel using the H_∞ criterion as well. The authors claim that their resulting minimax estimation method is highly robust and more appropriate in practical speech enhancement. It should be noted, however, that the implementation of the minimax criterion in the parameter estimation stage of the algorithm seems to be much more complicated than the conventional estimation procedure.

Wan et al. [8.36] assume a nonlinear model of the speech production, i. e., that the speech utterance is the output of a neural network (NN) with unknown parameters. Their algorithm is comprised of iterations between parameter estimation and signal enhancement. The nonlinearity inherent to the NN is addressed by the application of the EKF. The recently proposed unscented transform (UT), suggested by *Julier* et al. [8.37], is a novel method for calculating first- and second-order statistics of a random variable undergoing a nonlinear transformation, that was used for constructing a Kalman filter for the nonlinear case. The resulting filter, named the unscented Kalman filter (UKF), was shown to be superior to the well-established EKF in many application of interest. *Wan* et al. [8.38] propose to replace the EKF in [8.36] by the UKF, resulting in improved performance. *Gannot* and *Moonen* [8.39] use the UKF in the speech enhancement application (as well as speech dereverberation), where the nonlinearity arises from the multiplication of the speech and the parameters. Their proposed method is only applied to a synthetic AR process. *Fong* and *God-*

sill [8.40] use the particle Kalman filter for the speech enhancement task. The speech signal gain is given a random walk model, while its partial correlation coefficients (PARCOR) [8.41] are given a constrained random walk model (as their absolute value must be smaller than 1). Monte Carlo filtering is applied for estimating these parameters, whereas a linear Kalman filter is applied in parallel for enhancing the speech signal.

A distinct family of algorithms, also using the Kalman filter, employs a training stage for estimation of the associated parameters. In this approach a hidden Markov model (HMM) for the clean signals is estimated from the training data, and the clean signal is estimated from the noisy signal by applying Bayesian estimators. This method was first proposed by *Ephraim* et al. [8.42, 43], where a bank of HMM state-related Wiener filters was used. The Wiener filter was later replaced by the Kalman filter in [8.44, 45]. The problem of unknown speech gain contours and noise parameters is alleviated by using EM iterations. This method, proposed by *Lee* and *Jung* [8.46], use Kalman filter in the E-step, based on the trained AR parameters and the estimated noise parameters. In the M-step the noise and gain parameters are recursively estimated. An interacting multiple model (IMM) algorithm, in which the Kalman filters in the different states interact with one another, is applied for enhancing speech contaminated by additive white or colored noise by *Kim* et al. [8.47]. Finally, a nonlinear extension of the HMM concept is proposed by *Lee* et al. [8.48]. The speech is assumed to be an output of a NN with time-varying parameters controlled by a hidden Markov chain. Both the training and the enhancement stage become nonlinear. The nonlinearity is addressed by application of the EKF.

8.4.2 Speech Enhancement

In the following derivation we model both the speech and noise signals as AR processes. Hence, the case of temporally correlated noise, derived in Sect. 8.2.4, is considered. Note, however, that the speech and noise parameters are not known and should be estimated. We propose the framework of the EM algorithm for jointly estimating the signals and their parameters. For the application of EM we have to specify the probability distribution of the processes involved. In order to obtain a tractable estimation scheme, we shall assume Gaussian distribution models. Recall that in a Gaussian framework the Kalman filter's output is not only the optimal linear estimate, but also the general optimal estimate, namely, $\hat{\boldsymbol{x}}_{n|n} = E(\boldsymbol{x}_n|\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$.

For the full derivation of the related EM estimation scheme the reader is referred to [8.49].

The Signal Model

Consider a speech signal received by a single microphone and contaminated by a colored noise signal. Let the signal measured by the microphone be given by:

$$y(n) = x(n) + v(n)\,, \tag{8.97}$$

where $x(n)$ represents the sampled speech signal, and $v(n)$ represents additive background noise. We assume the standard LPC modeling for the speech signal over an analysis frame; i. e., $x(n)$ is modeled as a Gaussian stochastic AR process:

$$x(n) = -\sum_{k=1}^{p} \alpha_k x(n-k) + w^{\mathrm{s}}(n)\,, \tag{8.98}$$

where the excitation $w^{\mathrm{s}}(n)$ is a zero-mean white Gaussian noise with variance $E\{[w^{\mathrm{s}}(n)]^2\} = \sigma^2_{w^{\mathrm{s}}}$, and $\alpha_1, \ldots, \alpha_p$ are the AR coefficients, assumed to be time invariant over an analysis frame, due to the quasistationarity assumption. We may incorporate the more-detailed voiced speech model suggested in [8.50] in which the excitation process is composed of a weighted linear combination of an impulse train and a white-noise sequence to represent voiced and unvoiced speech, respectively. However, in our experiments, this approach did not yield any significant performance improvements over the standard LPC modeling. In reformulating (8.98) in the state-space representation, we take an approach that slightly differs from the state-space formulation of an AR process, defined in Sect. 8.2. The state vector, defined here, is a $(p+1)\times 1$ vector. The extra term will allow for more-convenient parameter estimation. Define

$$\boldsymbol{x}^{\mathrm{s}}_n = (x(n-p)\; x(n-p+1)\; \ldots\; x(n))^{\mathrm{T}}\,.$$

Define also the $(p+1)\times(p+1)$ speech transition matrix

$$\boldsymbol{\Phi}^{\mathrm{s}} = \begin{pmatrix} 0 & 1 & 0 & & \cdots & 0 \\ \vdots & 0 & \ddots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \ddots & \vdots \\ \vdots & \vdots & & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & \cdots & 0 & 1 \\ 0 & -\alpha_p & -\alpha_{p-1} & \cdots & -\alpha_2 & -\alpha_1 \end{pmatrix}\,.$$

Let $\boldsymbol{w}^{\mathrm{s}}_n$ be the $(p+1)\times 1$ driving noise vector and $\mathbf{H}^{\mathrm{s}}$ be the $1\times(p+1)$ the measurement matrix

$$\boldsymbol{w}^{\mathrm{s}}_n = (0\; \ldots\; 0\; w^{\mathrm{s}}(n))^{\mathrm{T}}\,,$$
$$\mathbf{H}^{\mathrm{s}} = (0\; \ldots\; 0\; 1)\,.$$

Then (8.98) can be rewritten as

$$\begin{aligned} \boldsymbol{x}_n^{\mathrm{s}} &= \boldsymbol{\Phi}^{\mathrm{s}} \boldsymbol{x}_{n-1}^{\mathrm{s}} + \boldsymbol{w}_n^{\mathrm{s}} \,, \\ y(n) &= \mathbf{H}^{\mathrm{s}} \boldsymbol{x}_n^{\mathrm{s}} + v(n) \,. \end{aligned} \tag{8.99}$$

The additive noise $v(n)$ is also assumed to be a realization from a zero-mean, possibly nonwhite Gaussian AR process:

$$v(n) = -\sum_{k=1}^{q} \beta_k v(n-k) + w^{\mathrm{v}}(n) \,, \tag{8.100}$$

where $\beta_1, \ldots, \beta_q$ are the AR parameters of the noise process, and the excitation $w^{\mathrm{v}}(n)$ is a white Gaussian noise with variance $E\{[w^{\mathrm{v}}(n)]^2\} = \sigma_{w^{\mathrm{v}}}^2$. Many common noise sources may indeed be closely approximated as low-order AR processes. Significant improvement may be obtained by incorporating the colored-noise model into the estimation process [8.15, 30].

Equation (8.100) can be rewritten in a state-space formulation as well. Define the $(q+1)\times 1$ state vector $\boldsymbol{x}_n^{\mathrm{v}} = (v(n-q) \;\; v(n-q+1) \;\; \ldots \;\; v(n))^{\mathrm{T}}$ and the $(q+1)\times(q+1)$ noise transition matrix

$$\boldsymbol{\Phi}^{\mathrm{v}} = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & & \ddots & \ddots & \vdots \\ \vdots & \vdots & & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & \cdots & 0 & 1 \\ 0 & -\beta_q & -\beta_{q-1} & \cdots & -\beta_2 & -\beta_1 \end{pmatrix} .$$

Let $\boldsymbol{w}_n^{\mathrm{v}}$ be the $(q+1)\times 1$ driving noise vector and $\mathbf{H}^{\mathrm{v}}$ be the $1\times(q+1)$ measurement vector

$$\begin{aligned} \boldsymbol{w}_n^{\mathrm{v}} &= (0 \;\ldots\; 0 \; w^{\mathrm{v}}(n))^{\mathrm{T}} \,, \\ \mathbf{H}^{\mathrm{v}} &= (0 \;\ldots\; 0 \; 1) \,. \end{aligned}$$

Using these definitions (8.100) can be rewritten as

$$\begin{aligned} \boldsymbol{x}_n^{\mathrm{v}} &= \boldsymbol{\Phi}^{\mathrm{v}} \boldsymbol{x}_{n-1}^{\mathrm{v}} + \boldsymbol{w}_n^{\mathrm{v}} \,, \\ y(n) &= s(n) + \mathbf{H}^{\mathrm{v}} \boldsymbol{x}_n^{\mathrm{v}} \,. \end{aligned} \tag{8.101}$$

Equations (8.99) and (8.101) can be concatenated into a one, larger, state-space equation.

$$\begin{aligned} \boldsymbol{x}_n &= \boldsymbol{\Phi} \boldsymbol{x}_{n-1} + \boldsymbol{w}_n \,, \\ y(n) &= \mathbf{H} \boldsymbol{x}_n \,. \end{aligned} \tag{8.102}$$

The augmented $(p+q+2)\times 1$ state vector $\boldsymbol{x}_n$ is defined by

$$\begin{aligned} \boldsymbol{x}_n^{\mathrm{T}} &= \left((\boldsymbol{x}_n^{\mathrm{s}})^{\mathrm{T}} \; (\boldsymbol{x}_n^{\mathrm{v}})^{\mathrm{T}} \right) \\ &= \left((\bar{\boldsymbol{x}}_n^{\mathrm{s}})^{\mathrm{T}} \; x(n) \; (\bar{\boldsymbol{x}}_n^{\mathrm{v}})^{\mathrm{T}} \; v(n) \right) , \end{aligned} \tag{8.103}$$

where $\bar{\boldsymbol{x}}_n^{\mathrm{s}}$ and $\bar{\boldsymbol{x}}_n^{\mathrm{s}}$ are $p\times 1$ and $q\times 1$ regression vectors, respectively, defined as

$$\begin{aligned} \bar{\boldsymbol{x}}_n^{\mathrm{s}} &= (x(n-p) \; x(n-p+1) \; \ldots \; x(n-1))^{\mathrm{T}} \,, \\ \bar{\boldsymbol{x}}_n^{\mathrm{v}} &= (v(n-q) \; v(n-q+1) \; \ldots \; v(n-1))^{\mathrm{T}} \,. \end{aligned}$$

The augmented state-transition matrix $\boldsymbol{\Phi}$ is given by

$$\boldsymbol{\Phi} = \begin{pmatrix} \boldsymbol{\Phi}^{\mathrm{s}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi}^{\mathrm{v}} \end{pmatrix} .$$

The driving noise vector is given by

$$\boldsymbol{w}_n^{\mathrm{T}} = \left((\boldsymbol{w}_n^{\mathrm{s}})^{\mathrm{T}} \; (\boldsymbol{w}_n^{\mathrm{v}})^{\mathrm{T}} \right)$$

and the measurement vector is given by

$$\mathbf{H} = (\mathbf{H}^{\mathrm{s}} \;\; \mathbf{H}^{\mathrm{v}}) \,.$$

The respective driving noise covariance matrices are given by the $(p+1)\times(p+1)$ matrix $\mathbf{Q}^{\mathrm{s}}$ and the $(q+1)\times(q+1)$ matrix $\mathbf{Q}^{\mathrm{v}}$ as

$$\mathbf{Q}^{\mathrm{s}} = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & \sigma_{w^{\mathrm{s}}}^2 \end{pmatrix}$$

$$\mathbf{Q}^{\mathrm{v}} = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & \sigma_{w^{\mathrm{v}}}^2 \end{pmatrix}$$

and the augmented covariance matrix

$$\mathbf{Q} = \begin{pmatrix} \mathbf{Q}^{\mathrm{s}} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}^{\mathrm{v}} \end{pmatrix} .$$

Note that the augmented state-space representation (8.102) is noise free. The signal $v(n)$ is treated as another source signal, rather than as a noise signal.

Assuming that all signal and noise parameters are known, which implies that $\boldsymbol{\Phi}$, $\mathbf{H}$, and $\mathbf{Q}$ are known, the optimal MMSE linear state estimate of $\boldsymbol{x}_n$ (consisting of the desired speech signal $s(n)$), given the frame samples $\{y(0), y(1), \ldots, y(N-1)\}$, is obtained by using the Kalman smoothing equations. However, since the signal and noise parameters are not known a priori, they must be estimated within the algorithm as well. If the casual Kalman filter is used, rather than the Kalman smoother, only a suboptimal estimation is obtained, but the performance loss is usually not too significant.

EM-Based Algorithm

The following (simplified) iterative algorithm was derived in [8.30, 49]. The algorithm that iterates between state and parameter estimation can be viewed as a simplified version of the EM [8.12] procedure for the problem at hand.

Let $\boldsymbol{\theta}$ be the vector of unknown parameters in the extended model

$$\boldsymbol{\theta}^{\mathrm{T}} = \left(\boldsymbol{\alpha}^{\mathrm{T}}\ \sigma^2_{w^{\mathrm{s}}}\ \boldsymbol{\beta}^{\mathrm{T}}\ \sigma^2_{w^{\mathrm{v}}}\right), \tag{8.104}$$

where

$$\boldsymbol{\alpha}^{\mathrm{T}} = (\alpha_p\ \alpha_{p-1}\ \ldots\ \alpha_1),$$
$$\boldsymbol{\beta}^{\mathrm{T}} = (\beta_q\ \beta_{q-1}\ \ldots\ \beta_1).$$

The EM algorithm is an iterative procedure for computing the ML estimate of the parameters. In our application, the noisy signal $y(n)$ is divided into non-overlapping frames comprising the samples $\{y(0), y(1)\ldots, y(N-1)\}$, where N is the frame length. The EM algorithm is applied to each frame separately. Denote by ℓ the iteration index in each frame. Let $\hat{\boldsymbol{\theta}}^{(\ell)}$ be the estimate of $\boldsymbol{\theta}$ after ℓ iterations of the algorithm in the current frame. Define, $\boldsymbol{y}_0^n = \{y(0), y(1), \ldots, y(n)\}$, the samples available for casual estimation. We use the notations

$$\hat{\boldsymbol{x}}_{n|n} = E_{\hat{\boldsymbol{\theta}}^{(\ell)}}\left(\boldsymbol{x}_n|\boldsymbol{y}_0^n\right) \tag{8.105}$$
$$\widehat{\boldsymbol{x}_n\boldsymbol{x}_n^{\mathrm{T}}} = E_{\hat{\boldsymbol{\theta}}^{(\ell)}}\left(\boldsymbol{x}_n\boldsymbol{x}_n^{\mathrm{T}}|\boldsymbol{y}_0^n\right)$$

for designating estimates of the first- and second-order statistics of the state vector, based on observations $\boldsymbol{y}_0^n$, and using the current parameter estimate $\hat{\boldsymbol{\theta}}^{(\ell)}$. In the Gaussian case, the term $\hat{\boldsymbol{x}}_{n|n}$ of (8.105) is exactly the output of the Kalman filter (operating with the parameters set $\hat{\boldsymbol{\theta}}^{(\ell)}$) bearing the same notation. Note, however, that in the non-Gaussian case the Kalman filter's output is merely the optimal *linear* estimate of $\boldsymbol{x}_n$ from $\boldsymbol{y}_0^n$, which differ from the conditional mean, namely from the optimal *general* estimate used in (8.105). The second-moment term $\widehat{\boldsymbol{x}_n\boldsymbol{x}_n^{\mathrm{T}}}$ is not directly available from the Kalman filter, however it may be obtained indirectly as follows.

The two terms in (8.105) are related via the corresponding conditional covariance matrix

$$\widehat{\boldsymbol{x}_n\boldsymbol{x}_n^{\mathrm{T}}} - \hat{\boldsymbol{x}}_{n|n}\hat{\boldsymbol{x}}_{n|n}^{\mathrm{T}} = E_{\hat{\boldsymbol{\theta}}^{(\ell)}}\left(\hat{\boldsymbol{x}}_{n|n} - \boldsymbol{x}_n\right) \cdot (\hat{\boldsymbol{x}}_{n|n} - \boldsymbol{x}_n)^{\mathrm{T}}|\boldsymbol{y}_0^n\big). \tag{8.106}$$

Recall that the Kalman filter's error covariance $\mathbf{P}_{n|n}$ is essentially the unconditional MSE

$$\mathbf{P}_{n|n} = E_{\hat{\boldsymbol{\theta}}^{(\ell)}}\left[(\hat{\boldsymbol{x}}_{n|n} - \boldsymbol{x}_n)(\hat{\boldsymbol{x}}_{n|n} - \boldsymbol{x}_n)^{\mathrm{T}}\right], \tag{8.107}$$

Luckily, however, in the Gaussian case the conditional MSE of the optimal estimate $\boldsymbol{x}_{n|n}$ does not depend on the data $\boldsymbol{y}_0^n$. This implies, by taking the mean of the conditional MSE with respect to $\boldsymbol{y}_0^n$, that the conditional MSE equals the (constant) unconditional MSE. We therefore conclude that the conditional covariance in (8.106) is given by the Kalman filter's output MSE $\mathbf{P}_{n|n}$.

Hence, the following relation can be used (in the Gaussian case) for estimating the second-order statistics of the state vector based on the observations $\boldsymbol{y}_0^n$ and the current parameter estimate

$$\widehat{\boldsymbol{x}_n\boldsymbol{x}_n^{\mathrm{T}}} = \mathbf{P}_{n|n} + \hat{\boldsymbol{x}}_{n|n}\hat{\boldsymbol{x}}_{n|n}^{\mathrm{T}}. \tag{8.108}$$

The terms $\hat{\boldsymbol{x}}_{n|n}$ and $\mathbf{P}_{n|n}$ will be calculated in the sequel using a forward Kalman filtering recursion.

To obtain the parameter estimation in the next iteration, $\hat{\theta}^{(\ell+1)}$, we use the following EM-based two-stage iterative procedure. The Kalman filter constitutes the estimation stage (also known as the E-step) of the EM algorithm. The Yule–Walker-based parameter estimation constitutes the maximization stage of the algorithm, known as the M-step.

E-step:

For $n = 0, 1, 2, \ldots, N-1$:

Propagation equations:

$$\hat{\boldsymbol{x}}_{n|n-1} = \hat{\boldsymbol{\Phi}}^{(\ell)}\hat{\boldsymbol{x}}_{n-1|n-1}$$
$$\mathbf{P}_{n|n-1} = \hat{\boldsymbol{\Phi}}^{(\ell)}\mathbf{P}_{n-1|n-1}(\hat{\boldsymbol{\Phi}}^{(\ell)})^{\mathrm{T}} + \hat{\mathbf{Q}}^{(\ell)}$$

Update equations:

$$\hat{\boldsymbol{x}}_{n|n} = \hat{\boldsymbol{x}}_{n|n-1} + \boldsymbol{k}_n[y(n) - \mathbf{H}\hat{\boldsymbol{x}}_{n|n-1}]$$
$$\mathbf{P}_{n|n} = \mathbf{P}_{n|n-1} - \boldsymbol{k}_n\mathbf{H}\mathbf{P}_{n|n-1}$$

Kalman gain:

$$\boldsymbol{k}_n = \frac{1}{\mathbf{H}\mathbf{P}_{n|n-1}\mathbf{H}^{\mathrm{T}}}\mathbf{P}_{n|n-1}\mathbf{H}^{\mathrm{T}}$$

$\hat{\boldsymbol{\Phi}}^{(\ell)}$ and $\hat{\mathbf{Q}}^{(\ell)}$ are estimates of the matrices $\boldsymbol{\Phi}$ and $\mathbf{Q}$, respectively, at the current iteration stage ℓ.

The estimated parameters in the next iteration $\ell+1$ are computed by using equation sets similar to the standard Yule–Walker solution for estimating the coefficients of an AR process, except that the correlation values are replaced by their a posteriori empirical value. This calculation constitutes the M-step of the algorithm.

M-step:

$$\hat{\boldsymbol{\alpha}}^{(l+1)} = -\left[\sum_{n=0}^{N-1} \widehat{\bar{\boldsymbol{x}}_n^{\mathrm{s}}(\bar{\boldsymbol{x}}_n^{\mathrm{s}})^{\mathrm{T}}}\right]^{-1} \sum_{n=0}^{N-1} \widehat{\bar{\boldsymbol{x}}_n^{\mathrm{s}} x(n)}$$

$$\widehat{\sigma_{w^{\mathrm{s}}}^2}^{(l+1)} = \frac{1}{N}\sum_{n=0}^{N-1}\left[\widehat{x^2(n)} + (\hat{\boldsymbol{\alpha}}^{(l+1)})^{\mathrm{T}}\widehat{\bar{\boldsymbol{x}}_n^{\mathrm{s}} x(n)}\right]$$

$$\hat{\boldsymbol{\beta}}^{(l+1)} = -\left[\sum_{n=0}^{N-1} \widehat{\bar{\boldsymbol{x}}_n^{\mathrm{v}}(\bar{\boldsymbol{x}}_n^{\mathrm{v}})^{\mathrm{T}}}\right]^{-1} \sum_{n=0}^{N-1} \widehat{\bar{\boldsymbol{x}}_n^{\mathrm{v}} v(n)}$$

$$\widehat{\sigma_{w^{\mathrm{v}}}^2}^{(l+1)} = \frac{1}{N}\sum_{n=0}^{N-1}\left[\widehat{v^2(n)} + (\hat{\boldsymbol{\beta}}^{(l+1)})^{\mathrm{T}}\widehat{\bar{\boldsymbol{x}}_n^{\mathrm{v}} v(n)}\right]$$

We note that $\widehat{\bar{\boldsymbol{x}}_n^{\mathrm{s}}(\bar{\boldsymbol{x}}_n^{\mathrm{s}})^{\mathrm{T}}}$ is the upper left $p\times p$ submatrix of $\widehat{\boldsymbol{x}_n\boldsymbol{x}_n^{\mathrm{T}}}$. $\widehat{\bar{\boldsymbol{x}}_n^{\mathrm{s}} x(n)}$, $\widehat{x^2(n)}$, $\widehat{\bar{\boldsymbol{x}}_n^{\mathrm{v}}(\bar{\boldsymbol{x}}_n^{\mathrm{v}})^{\mathrm{T}}}$, $\widehat{\bar{\boldsymbol{x}}_n^{\mathrm{v}} v(n)}$, and $\widehat{v^2(n)}$ may similarly be extracted from $\widehat{\boldsymbol{x}_n\boldsymbol{x}_n^{\mathrm{T}}}$. In fact, this is the reason for choosing $p+1$- and $q+1$-dimensional state vectors for this estimation scheme, although, as seen in Sect. 8.2, p- and q-dimensional state vectors are sufficient for Kalman filtering when the parameters are known.

Although the forward Kalman filter was used instead of the optimal Kalman smoother, it is empirically shown that the EM-based method maintains a monotonic convergence behavior towards the ML estimate of all unknown parameters or at least to a local maximum of the likelihood function. Each iteration increases the likelihood of the estimate of the parameters and improves the signal state-space estimator in MMSE sense.

The iterative algorithm [designated the Kalman EM iterative (KEMI) method in [8.30]] is summarized in Fig. 8.2. Note, that two other blocks are depicted in the figure. The first block, denoted *framing*, is responsible for the segmentation of the noisy speech signal. The second block is the *initialization* block. The initialization procedure, based on the use of HOS, is beyond the scope of this summary.

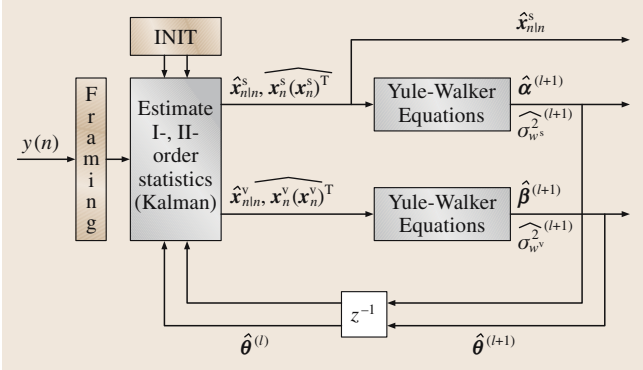


Fig. 8.2 Iterative Kalman algorithm based on the EM procedure

Demonstration

The performance of the KEMI algorithm is demonstrated using the sonograms depicted in Fig. 8.3. We used a speech signal drawn from TIMIT [8.51] and downsampled to 8 kHz. The speech signal was degraded by additive noise at SNR = 5 dB. The noise source was the speech-like noise signals drawn from the NOISEX-92 [8.52] database. In applying the algorithm we used an AR model with order $p = 10$ for the speech signal and for modeling the nonwhite noise signal we used an AR model with order $q = 4$. In this experiment, we used five iterations.

It is clearly shown in Fig. 8.3 that the noise signal is attenuated while imposing only minor distortion on the speech signal. For a more-comprehensive and comparative experimental study, including objective distortion measures as well as recognition rate of a speech recognition algorithm, please refer to [8.49].

8.4.3 Speaker Tracking

Determining the spatial position of a speaker finds a growing interest in video conference scenarios where automated camera steering and tracking are required. Acoustic source localization might also be used as a preprocessor stage for speech enhancement algorithms, which are based on microphone array beamformers.

Usually, methods for speaker localization consist of two stages. In the first stage, a microphone array is used to extract the time difference between arrivals of the speech signal at each pair of microphones. For each pair the time difference of arrival (TDOA) is estimated using spatially separated microphone pairs. A classical method for estimating the TDOA is the generalized cross-correlation (GCC) algorithm [8.53].

In the second stage the noisy TDOA readings from all pairs are combined in order to produce the source location estimate. In a dynamic setting the source location may be time-varying, thus its estimate must be constantly updated (tracked) as new TDOA readings become available. It has become common practice to employ Kalman filtering in such tracking problems, so as to properly weigh the effect of incoming TDOA readings on the estimated source location, accounting for a smooth propagation model of the source's trajectory (when applicable). Although not directly related to the processing of the speech signal itself, the use of Kalman

filtering in this problem serves to demonstrate the diversity and versatility of this remarkable and practical estimation tool.

Algorithm Derivation

Consider an $M+1$-microphone array as depicted in Fig. 8.4.

The microphones are placed at the Cartesian coordinates $\boldsymbol{m}_i = (x_i \;\; y_i \;\; z_i)^{\mathrm{T}}$; $i = 0, \ldots, M$. To simplify the exposition, the location of a reference microphone $\boldsymbol{m}_0$ is set at the origin of the axes $\boldsymbol{m}_0 = (0 \;\; 0 \;\; 0)^{\mathrm{T}}$. Define the source coordinates at time instant n by $\boldsymbol{s}_n = (x_{\mathrm{s}}(n) \;\; y_{\mathrm{s}}(n) \;\; z_{\mathrm{s}}(n))^{\mathrm{T}}$. Each of the M microphones, combined with the reference microphone, is used at time instant n to extract a TDOA measurement $\tau_i(n)$; $i = 1, \ldots, M$. Denote the i-th range difference measurement by $r_i(n) = c\tau_i(n)$, where c is the sound propagation speed (approximately 340 m/s in air). It can easily be verified from simple geometrical considerations (see Fig. 8.4) that this range difference is related to the source and the microphone location by the nonlinear equation

$$r_i(n) = \|\boldsymbol{s}_n - \boldsymbol{m}_i\| - \|\boldsymbol{s}_n\|; \; i = 1, \ldots, M\,, \qquad (8.109)$$

where $\|\boldsymbol{s}_n\| = \sqrt{x_s^2(n) + y_s^2(n) + z_s^2(n)}$ denote the Euclidean norm of the source coordinates.

Usually, only estimates of the true TDOAs are available. Thus, concatenating M estimates of the quantity in (8.109), a nonlinear measurement model is obtained:

$$\boldsymbol{r}_n = \begin{pmatrix} \|\boldsymbol{s}_n - \boldsymbol{m}_1\| - \|\boldsymbol{s}_n\| \\ \vdots \\ \|\boldsymbol{s}_n - \boldsymbol{m}_M\| - \|\boldsymbol{s}_n\| \end{pmatrix} + \boldsymbol{v}_n = \boldsymbol{h}(\boldsymbol{s}_n) + \boldsymbol{v}_n\,. \qquad (8.110)$$

Here, $\boldsymbol{v}_n^{\mathrm{T}} = (v_1(n) \;\; v_2(n) \;\; \ldots \;\; v_M(n))$ is a vector of estimation errors with zero mean and covariance $\mathbf{R}_n = E(\boldsymbol{v}_n \boldsymbol{v}_n^{\mathrm{T}})$. The goal of the localization task is to extract the speaker's trajectory $\boldsymbol{s}_n$ from the measurements vector $\boldsymbol{r}_n$. Any estimation procedure (e.g., [8.53]) could be used for the TDOA estimation. The method introduced in the sequel, constituting the second stage of the localization procedure, is independent of the choice of the first stage.

We propose a dynamic model for the source trajectory. As the actual track is unknown, a simplified random walk model is used instead.

$$\boldsymbol{s}_{n+1} = \boldsymbol{\Phi}\boldsymbol{s}_n + \boldsymbol{w}_n\,, \qquad (8.111)$$

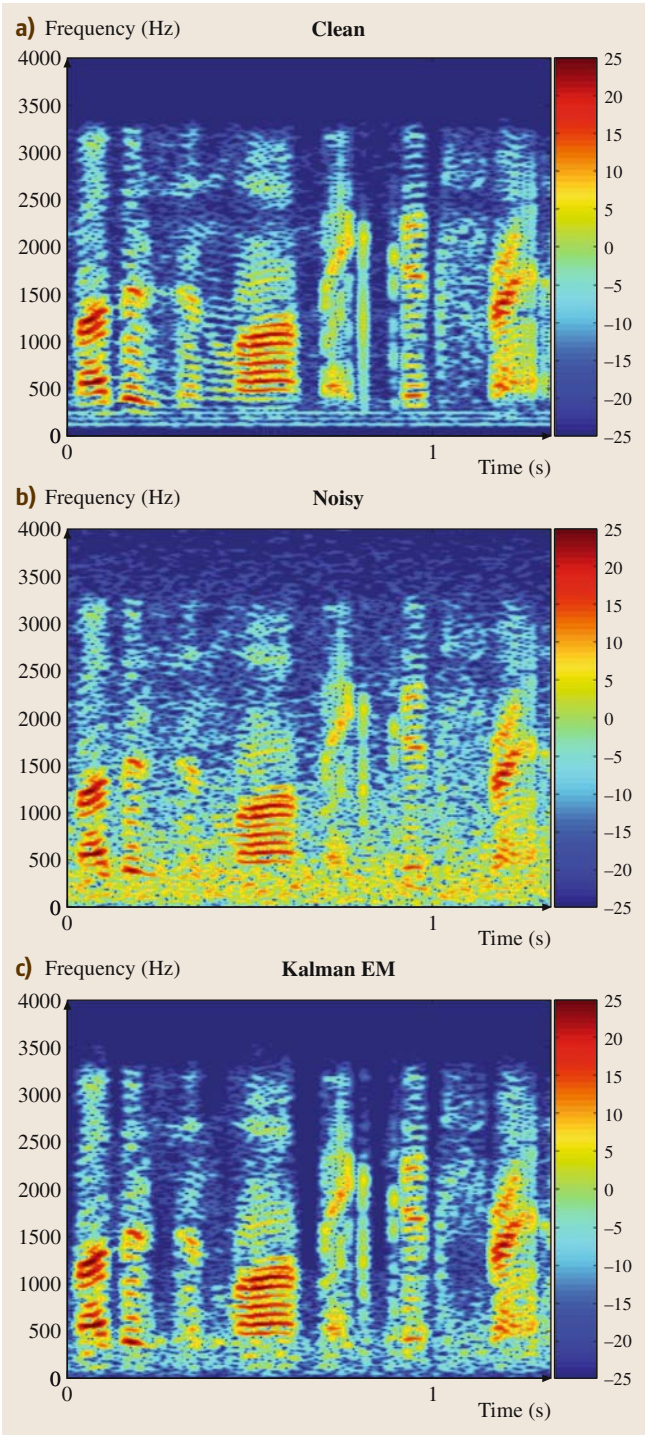


Fig. 8.3a–c Iterative Kalman algorithm based on the EM procedure

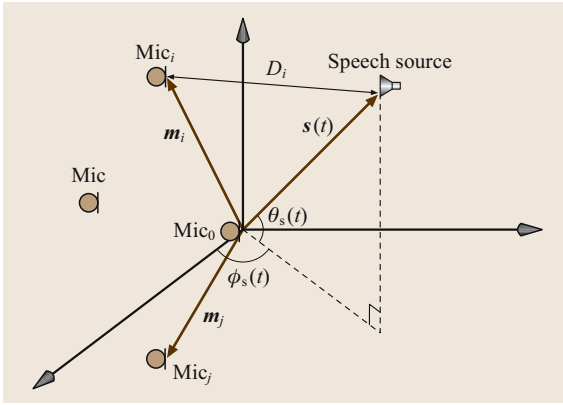


Fig. 8.4 Microphone array. Speaker location at time instant n is $\boldsymbol{s}_n$ with azimuth angle $\phi_s(n)$ and elevation angle $\theta_s(n)$. Microphone position notated by $\boldsymbol{m}_i$; $i = 0, \ldots, M$

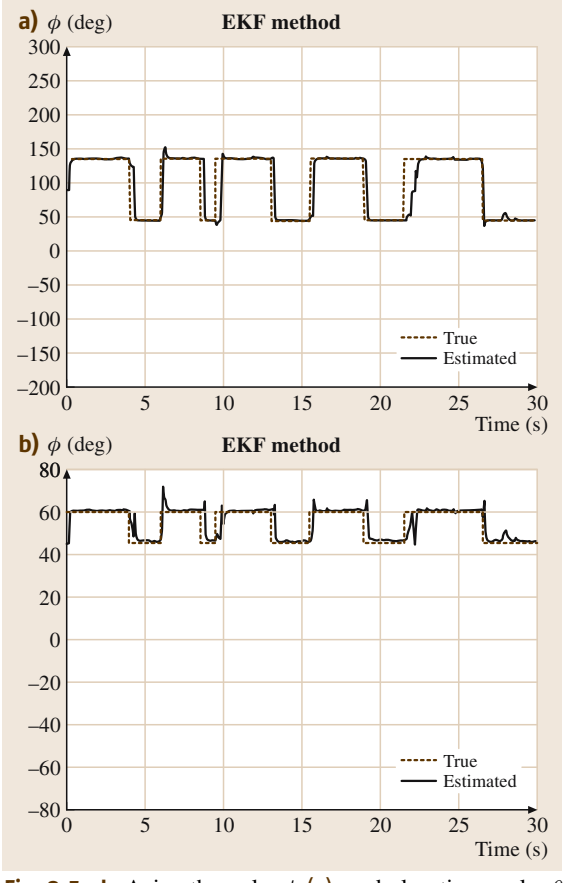


Fig. 8.5a,b Azimuth angle, ϕ (**a**), and elevation angle, θ (**b**) estimation results in a switching scenario

where $\boldsymbol{w}_n$ is the coordinate-wise, zero-mean, temporally white driving noise with covariance matrix $\mathbf{Q}_n = E(\boldsymbol{w}_n \boldsymbol{w}_n^{\mathrm{T}})$. $\boldsymbol{\Phi}$ is a transition matrix assumed to be close to the identity matrix. This dynamic model together with the nonlinear measurement model given in (8.110) constitutes the state-space equation.

Since this model is nonlinear (due to the measurement equation) the classical Kalman filter cannot be used to estimate the state vector. Hence, a nonlinear extension thereof is required. We propose to use the EKF, derived in Sect. 8.3.3. This procedure only yields a suboptimal solution to the problem at hand. We provide here, for the completeness of the exposition, the calculations involved in the EKF aiming to solve the localization problem. Note, that in our case, (8.111) is already linear. However the measurement model in (8.110) still needs to be linearized. Assume that an estimate $\hat{\boldsymbol{s}}_{n-1|n-1}$ of the speaker location at time instant $n-1$, is known, as well as its corresponding error covariance matrix, $\mathbf{P}_{n-1|n-1}$. Then, recalling that the model equation is linear, the EKF recursion takes the following form.

Propagation equations:

$$\hat{\boldsymbol{s}}_{n|n-1} = \boldsymbol{\Phi}\hat{\boldsymbol{s}}_{n-1|n-1} \tag{8.112a}$$

$$\mathbf{P}_{n|n-1} = \boldsymbol{\Phi}\mathbf{P}_{n-1|n-1}\boldsymbol{\Phi}^{\mathrm{T}} + \mathbf{Q}_n \tag{8.112b}$$

Update equations:

$$\hat{\boldsymbol{s}}_{n|n} = \hat{\boldsymbol{s}}_{n|n-1} + \mathbf{K}_n(\boldsymbol{r}_n - \boldsymbol{h}(\hat{\boldsymbol{s}}_{n|n-1})) \tag{8.112c}$$

$$\mathbf{H}_n = \nabla_{\boldsymbol{s}_n}\boldsymbol{h}(\hat{\boldsymbol{s}}_{n|n-1}) \tag{8.112d}$$

$$= \begin{pmatrix} \left(\frac{\hat{\boldsymbol{s}}_{n|n-1}-\boldsymbol{m}_1}{\|\hat{\boldsymbol{s}}_{n|n-1}-\boldsymbol{m}_1\|} - \frac{\hat{\boldsymbol{s}}_{n|n-1}}{\|\hat{\boldsymbol{s}}_{n|n-1}\|}\right)^{\mathrm{T}} \\ \vdots \\ \left(\frac{\hat{\boldsymbol{s}}_{n|n-1}-\boldsymbol{m}_M}{\|\hat{\boldsymbol{s}}_{n|n-1}-\boldsymbol{m}_M\|} - \frac{\hat{\boldsymbol{s}}_{n|n-1}}{\|\hat{\boldsymbol{s}}_{n|n-1}\|}\right)^{\mathrm{T}} \end{pmatrix}$$

$$\mathbf{P}_{n|n} = (\mathbf{I} - \mathbf{K}_n\mathbf{H}_n)\,\mathbf{P}_{n|n-1} \tag{8.112e}$$

Kalman gain:

$$\mathbf{K}_n = \mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}}\left(\mathbf{H}_n\mathbf{P}_{n|n-1}\mathbf{H}_n^{\mathrm{T}} + \mathbf{R}_n\right)^{-1} \tag{8.112f}$$

with the initialization $\hat{\boldsymbol{s}}_{0|0}$ and its respective covariance $P_{0|0}$.

Demonstration

Consider the following simulation, which is typical of a video conference scenario. Two speakers, located at two different and fixed locations, speak alternately. The camera should be able to maneuver from one person to the other. For this scenario,

simulation is conducted with one speaker located at the polar position ($\phi = \frac{\pi}{4}$ rad $\theta = \frac{\pi}{4}$ rad $R = 1.5$ m) and the other at ($\phi = \frac{3\pi}{4}$ rad $\theta = \frac{\pi}{3}$ rad $R = 1.5$ m). A directional interference is placed at the position ($\phi = \frac{\pi}{2}$ rad $\theta = \frac{\pi}{4}$ rad $R = 1.0$ m). Six microphones were mounted at the following positions (Cartesian coordinates, in meters), relative to the reference microphone (which is at the axes origin):

$$\begin{aligned} \boldsymbol{m}_1^{\mathrm{T}} &= (0.3 \quad 0 \quad 0), \quad \boldsymbol{m}_2^{\mathrm{T}} = (-0.3 \quad 0 \quad 0), \\ \boldsymbol{m}_3^{\mathrm{T}} &= (\ 0 \ 0.3 \quad 0), \quad \boldsymbol{m}_4^{\mathrm{T}} = (\ 0 \ -0.3 \quad 0), \\ \boldsymbol{m}_5^{\mathrm{T}} &= (\ 0 \quad 0 \ 0.3), \quad \boldsymbol{m}_6^{\mathrm{T}} = (\ 0 \quad 0 \ -0.3). \end{aligned} \tag{8.113}$$

We used TDOA readings estimated from the noisy microphones data by the RS1 algorithm described in [8.54]. For this estimation stage, room reverberation (set to a reverberation time of $T_{60} = 0.25$ s) and the directional interferer were taken into account in simulation. Room reverberation was simulated by the image method [8.55]. The mean SNR level was set to 10 dB. Figure 8.5 presents the azimuth and elevation angles estimates obtained by the EKF method. The tracking ability of the algorithm is evident, even with abrupt changes of the desired speaker location.

8.5 Summary

In this chapter, we have introduced one of the most important contributions to the field of statistical signal processing, namely, the Kalman filter. We provided a rigorous derivation of the filter, based on the orthogonality principle. We also introduced several important variants of the Kalman filter: various Kalman smoothers, the Kalman predictor, a nonlinear extension (the EKF), and the treatment of temporally correlated measurement noise. We showed that the Kalman filter can easily deal with quite common parametric families of processes (AR, MA, and ARMA).

Finally, we demonstrated the application of the Kalman filter to two important speech processing problems, namely, speech enhancement and speaker localization. Despite the fact that the Kalman filter has already been used for almost two decades in speech enhancement applications, there is still much to do to improve the performance of such Kalman-filter-based algorithms. The main advantage of these algorithms stems from the fact that the Kalman filter may be continuously updated. The speech obtained has a natural sound and the residual noise is clean of annoying artifacts. However, the obtained noise suppression seems limited. This disadvantage, in our opinion, stems from the linear processing regime in which the Kalman filter is applied. Incorporating nonlinearity might yield better noise reduction, while maintaining the low distortion and keeping the low computational load. In localization problems, however, the Kalman filter still finds a leading position, more than 40 years after it was first introduced.

References

8.1 M.S. Grewal, A.P. Andrews: *Kalman Filtering, Theory and Practice*, Information and System Sciences series (Prentice-Hall, New Jersey 1993)

8.2 R.E. Kalman: A new approach to linear filtering and prediction problems, ASME J. Basic Eng. **82**(Series D), 35–45 (1960)

8.3 R.E. Kalman: New methods and results in linear prediction and filtering theory, Proc. Symposium on Engineering Applications of Random Function Theory and Probability (Wiley, New York 1961)

8.4 S.F. Schmidt: *Computational techniques in Kalman filtering* (NATO Advisory Group for Aerospace Research and Development, London 1970)

8.5 S.F. Schmidt: *Practical Aspects of Kalman filtering implementation* (NATO Advisory Group for Aerospace Research and Development, London 1976)

8.6 P.S. Maybeck: *Stochastic Models, Estimation and Control*, Vol. 2 (Academic, San Diego 1982)

8.7 T. Kailath: *Lectures on Wiener and Kalman Filtering* (Springer, New York 1981)

8.8 S. Haykin (Ed.): *Kalman filtering and neural networks*, Adaptive and Learning Systems for Signal Processing (Wiley, New York 2001)

8.9 H.E. Rauch, F. Tung, C.T. Striebel: Maximum likelihood estimates of linear dynamic systems, AIAA J. **3**, 1445–1450 (1965)

8.10 L.A. McGee, S.F. Schmidt: *Discovery of the Kalman Filter as a Practical Tool for Aerospace and Industry*, National Aeronautics and Space Administration (NASA) Technical Memorandum 86847, Nov. 1985

8.11 N. Wiener: *The Extrapolation, Interpolation and Smoothing of Stationary Time Series* (Wiley, New York 1949)

8.12 A.P. Dempster, N.M. Laird, D.B. Rubin: Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. B **39**(1), 1–38 (1977)

8.13 K.K. Paliwal, A. Basu: A speech enhancement method based on Kalman filtering, Proc. IEEE ICASSP (IEEE, Dallas 1987) pp. 177–180

8.14 B. Koo, J.D. Gibson, S.D. Gray: Filtering of colored noise for speech enhancement and coding, Proc. IEEE ICASSP (IEEE, Glasgow 1989) pp. 349–352

8.15 J.D. Gibson, B. Koo, S.D. Gray: Filtering of colored noise for speech enhancement and coding, IEEE Trans. Acoust. Speech **39**(6), 1732–1742 (1991)

8.16 S. Haykin: *Adaptive Filter Theory*, Information and System Sciences, 4th edn. (Prentice-Hall, Upper Saddle River 2002)

8.17 M. Feder, A.V. Oppenheim, E. Weinstein: Methods for noise cancellation based on the EM algorithm, Proc. IEEE ICASSP (IEEE, Dallas 1987) pp. 201–204

8.18 B. Widrow, J.R. Glover Jr., J.M. McCool, J. Kaunitz, C.S. Williams, R.H. Hearn, J.R. Zeider, E. Dong Jr., R.C. Goodlin: Adaptive noise cancelling: principals and applications, Proc. IEEE **63**(12), 1692–1716 (1975)

8.19 M. Feder, A.V. Oppenheim, E. Weinstein: Maximum likelihood noise cancellation using the EM algorithm, IEEE Trans. Acoust. Speech **37**(2), 204–216 (1989)

8.20 E. Weinstein, A.V. Oppenheim, M. Feder: *Signal Enhancement Using Single and Multi-Sensor Measurements* (MIT, Cambridge 1990)

8.21 J.S. Lim, A.V. Oppenheim: All-pole modeling of degraded speech, IEEE Trans. Acoust. Speech **26**(3), 197–210 (1978)

8.22 E. Weinstein, A.V. Oppenheim, M. Feder, J.R. Buck: Iterative and sequential algorithms for multisensor signal enhancement, IEEE Trans. Signal Process. **42**(4), 846–859 (1994)

8.23 M. Feder, E. Weinstein, A.V. Oppenheim: A new class of sequential and adaptive algorithms with application to noise cancellation, Proc. IEEE ICASSP (IEEE, New York 1988) pp. 557–560

8.24 A.V. Oppenheim, E. Weinstein, K.C. Zangi, M. Feder, D. Gauger: Single-sensor active noise cancellation, IEEE Trans. Speech Audio Process. **2**(2), 285–290 (1994)

8.25 B.-G. Lee, K.Y. Lee, S. Ann: An EM-based approach for parameter enhancement with an application to speech signals, Signal Process. **46**, 1–14 (1995)

8.26 K.Y. Lee, B.-G. Lee, S. Ann: Adaptive filtering for speech enhancement in colored noise, IEEE Signal Process. Lett. **4**(10), 277–279 (1997)

8.27 Z. Goh, K.-C. Tan, B.T.G. Tan: Kalman-Filtering Speech Enhancement Method Based on a Voiced-Unvoiced Speech Model, IEEE Trans. Speech Audio Process. **7**(5), 510–524 (1999)

8.28 M. Gabrea, E. Grivel, M. Najim: A single microphone Kalman filter-based noise canceller, IEEE Signal Process. Lett. **6**(3), 55–57 (1999)

8.29 B.D.O. Anderson, J.B. Moore: *Optimal Filtering*, Information and System Sciences Series (Prentice-Hall, Englewood Cliffs 1979)

8.30 S. Gannot, D. Burshtein, E. Weinstein: Iterative and sequential Kalman filter-based speech enhancement algorithms, IEEE Trans. Speech Audio Process. **6**(4), 373–385 (1998)

8.31 M. Fujimoto, Y. Ariki: Noisy speech recognition using noise reduction method based nn Kalman filter, Proc. IEEE ICASSP (IEEE, Istanbul 2000) pp. 1727–1730

8.32 S.F. Boll: Suppression of acoustic noise in speech using spectral subtraction, IEEE Trans. Acoust. Speech **27**(2), 113–120 (1979)

8.33 K.Y. Lee, B.-G. Lee, I. Song, S. Ann: Robust estimation of ar parameters and its application for speech enhancement, Proc. IEEE ICASSP (IEEE, San Francisco 1992) pp. 309–312

8.34 N. Ma, M. Bouchard, R.A. Goubran: Perceptual Kalman filtering for speech enhancement in colored noise, Proc. IEEE ICASSP, Vol. 1 (IEEE, Montreal 2004) pp. 717–720

8.35 X. Shen, L. Deng: A dynamic system approach to speech enhancement using the H_∞ filtering algorithm, IEEE Trans. Speech Audio Process. **27**(4), 391–399 (1999)

8.36 E.A. Wan, A.T. Nelson: Removal of noise from speech using the dual Ekf algorithm, Proc. IEEE ICASSP (IEEE, Seattle, Washington 1998)

8.37 S.J. Julier, J.K. Uhlmann: Unscented filtering and nonlinear estimation, Proc. IEEE **92**(3), 401–422 (2004)

8.38 E.A. Wan, R. van der Merwe: The unscented Kalman filter for nonlinear estimation, Proceedings of Symposium 2000 on Adaptive Systems for Signal Processing, Communication and Control (AS-SPCC) (IEEE, Lake Louise 2000)

8.39 S. Gannot, M. Moonen: On the application of the unscented Kalman filter to sppech processing, The International Workshop on Acoustic Echo and Noise Control (IWAENC) (Kyoto 2003) pp. 27–30

8.40 W. Fong, S. Godsill: Monte Carlo smoothing with application to audio signal enhancement, IEEE SSP Workshop (IEEE, Singapore 2001) pp. 18–210

8.41 T.W. Parsons: *Voice and Speech Processing* (McGraw-Hill, USA 1987)

8.42 Y. Ephraim, D. Malah, B.H. Juang: On the application of hidden Markov models for enhancing noisy speech, IEEE Trans. Acoust. Speech **37**, 1846–1856 (1989)

8.43 Y. Ephraim: A bayesian estimation approach for speech enhancement using hidden markov models, IEEE Trans. Signal Process. **40**, 725–735 (1992)

8.44 Y. Ephraim: Speech enhancement using state dependent dynamical system model, Proc. IEEE ICASSP (IEEE, San Francisco 1992) pp. 289–292

8.45 K.Y. Lee, K. Shirai: Efficient recursive estimation for speech enhancement in colored noise, IEEE Signal Process. Lett. **3**, 196–199 (1996)

8.46 K.Y. Lee, S. Jung: Time-domain approach using multiple Kalman filters and em algorithm to speech

enhancement with nonstationary noise, IEEE Trans. Speech Audio Process. **8**(3), 373–385 (2000)

8.47 J.B. Kim, K.Y. Lee, C.W. Lee: On the applications of the interacting multiple model algorithm for enhancing noisy speech, IEEE Trans. Speech Audio Process. **8**(3), 349–352 (2000)

8.48 K.Y. Lee, S. McLaughlin, K. Shirai: Speech enhancement based on extended Kalman filter and neural predictive hidden Markov model, Proc. Int. Workshop Neural Networks for Signal Processing (IEEE, Kyoto 1996) pp. 302–310

8.49 S. Gannot: Speech Enhancement: Appliction of the Kalman filter in the estimate-maximize (EM) framework. In: *Speech Enhancement*, Signals and Communication Technology, ed. by S. Makino, J. Benesty, J. Chen (Springer, Berlin 2005), pp. 161–198

8.50 D. Burshtein: Joint modeling and maximum-likelihood estimation of pitch and linear prediction coefficient parameters, J. Acoust. Soc. Am. **3**, 1531–1537 (1992)

8.51 J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, N. L. Dahlgren, V. Zue: Acoustic-Phonetic Continuous Speech Corpus (TIMIT), CD-ROM (Oct. 1991)

8.52 A. Varga, H.J.M. Steeneken: Assessment for automatic speech recognition: II. NOISEX-92: a database and an experiment to study the effect of additive noise on speech recognition systems, Speech Commun. **12**, 247–251 (1993)

8.53 C.H. Knapp, G.C. Carter: The generalized correlation method for estimation of time delay, IEEE Trans. Acoust. Speech **24**(4), 320–327 (1976)

8.54 T.G. Dvorkind, S. Gannot: Time difference of arrival estimation of speech source in a noisy and reverberant environment, Signal Process. **85**(1), 177–204 (2005)

8.55 J.B. Allen, D.A. Berkley: Image method for efficiently simulating small-room acoustics, J. Acoust. Soc. Am. **65**(4), 943–950 (1979)

12. The STFT, Sinusoidal Models, and Speech Modification

M. M. Goodwin

Frequency-domain signal representations are used for a wide variety of applications in speech processing. In this Chapter, we first consider the short-time Fourier transform (STFT), presenting a number of interpretations of the analysis–synthesis process in a consistent mathematical framework. We then develop the sinusoidal model as a parametric extension of the STFT wherein the data in the STFT is compacted, sacrificing perfect reconstruction at the benefit of achieving a sparser and essentially more meaningful representation. We discuss several methods for sinusoidal parameter estimation and signal reconstruction, and present a detailed treatment of a matching pursuit algorithm for sinusoidal modeling. The final part of the Chapter addresses speech modifications such as filtering, enhancement, and time-scaling, for which both the STFT and the sinusoidal model are effective tools.

Frequency-domain representations have a rich history in applied mathematics and signal processing; their uses range from solving differential equations to compressing music. The cornerstone of frequency-domain approaches is the continuous-frequency Fourier transform of infinite-duration continuous-time signals; in the field of signal processing, this is of course an essential tool for the analysis of linear systems. For real-world digital signal processing applications, however, it is necessary to modify the Fourier transform to handle discrete-time signals of arbitrary duration in a segment-by-segment fashion, creating a *local* discrete-frequency Fourier representation for each segment. This idea of a short-time Fourier transform (STFT) was developed largely in the speech processing research community based on early work in signal processing and communications theory [12.1], for instance the key contributions by *Gabor* on time–frequency representations based on using modulated and time-shifted Gaussian windows to analyze the signal [12.2, 3]. Given this historical context, the STFT is sometimes referred to as a Gabor transform [12.4], although in some cases a distinction is drawn.

The short-time Fourier transform has proven useful for many speech processing and speech communication applications, including time scaling, pitch shifting, noise

reduction, and echo cancelation. Furthermore, development and understanding of the STFT led to advances in the theory and implementation of other types of transforms and filter banks, such as those that are now central to widely used image and audio compression schemes [12.4, 5]. In addition, a progression of ideas can be traced from the STFT and early vocoders to the sinusoidal model, which has been shown to be effective for low-rate speech and audio coding as well as for signal modification [12.6]. Of course, it should be noted that many of these frequency-domain processing schemes rely on the fast Fourier transform (FFT) and other similar algorithms, without which many of the implementations would not be computationally practical for real-world deployment [12.7, 8].

This chapter is organized as follows. Section 12.1 develops the short-time Fourier transform and interprets it in three different ways: as a sliding-window transform, as a modulated filter bank, and as a heterodyne filter bank. Section 12.2 then introduces the sinusoidal model as a parametric extension of the STFT. Methods for sinusoidal analysis and synthesis are discussed, with a focus on matching pursuit adapted for sinusoidal modeling. Section 12.3 provides an overview of the use of the STFT and sinusoidal models to realize desirable speech modifications such as filtering and time scaling.

12.1 The Short-Time Fourier Transform

The short-time Fourier transform (STFT) is a commonly used tool for analysis, modification, and synthesis of speech and audio signals. In this Section, the STFT is developed as a sliding-window transform, and conditions for perfectly reconstructing the input signal with an inverse transform and overlap-add are derived. It is then shown that the sliding-window transform corresponds to a modulated filter bank, and that the conditions for perfect reconstruction using a synthesis filter bank are equivalent to those for the sliding-window case. These formulations are related to the original definition of the STFT, which is shown to correspond to a heterodyne filter bank. The properties of the STFT are illustrated with examples, and some limitations of the STFT for processing real-world signals are described.

12.1.1 The STFT as a Sliding-Window Transform

In practical applications, the short-time Fourier transform is typically implemented in a sliding-window fashion. Given an input signal $x[t]$ of arbitrary duration, data segments are extracted at regular intervals using a time-limited window $w[n]$; these signal segments or frames can be expressed as

$$x_l[n] = w[n]x[n+lL], \quad 0 \le n \le N-1\,, \tag{12.1}$$

where N is the window length, l is a frame index, and L is the hop size, i. e., the spacing in samples between consecutive applications of the sliding extraction window; the index n is a local time index, i. e., an index relative to the start of the sliding window. For each signal frame, a discrete Fourier transform (DFT) is carried out, yielding

$$\begin{aligned} X[k,l] &= \sum_{n=0}^{N-1} x_l[n]\mathrm{e}^{-\mathrm{i}2\pi nk/K} \\ &= \sum_{n=0}^{N-1} w[n]x[n+lL]\mathrm{e}^{-\mathrm{i}2\pi nk/K}\,, \end{aligned} \tag{12.2}$$

where K is the DFT size and k is a frequency index or *bin* index. The STFT $X[k,l]$ then characterizes the local time–frequency behavior of the signal around time lL and bin k; for a sampling rate F_s, these discrete indices correspond to continuous time lL/F_s and frequency kF_s/K. To simplify the notation, a radial frequency

$$\omega_k = \frac{2\pi k}{K} \tag{12.3}$$

is often incorporated, such that the STFT expression becomes

$$X[k,l] = \sum_{n=0}^{N-1} w[n]x[n+lL]\mathrm{e}^{-\mathrm{i}\omega_k n}\,. \tag{12.4}$$

Note that for a hop size of $L = 1$, the STFT can be written directly as a function of the original time variable t:

$$X[k,t] = \sum_{n=0}^{N-1} w[n]x[n+t]\mathrm{e}^{-\mathrm{i}\omega_k n}\,. \tag{12.5}$$

Equations (12.4) and (12.5) are somewhat different from the formulation of the STFT in early references [12.9–12], as will be explored in Sect. 12.1.3.

In the sliding-window framework, the STFT is thought of as a spectral representation of a time slice

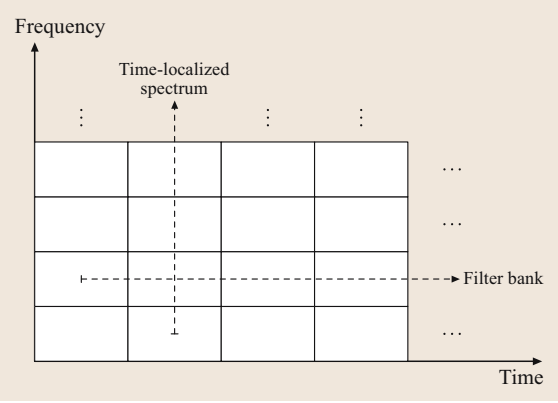


Fig. 12.1 Interpretations of the short-time Fourier transform as a series of time-localized spectra (*vertical*) and as a bank of bandpass filters (*horizontal*)

of the input signal; interpreting $X[k,l]$ as a function of the frequency k for each value of the time index l, the STFT corresponds to a series of time-localized spectra. Alternatively, the STFT can be viewed as a function of time for each frequency; interpreting $X[k,l]$ as a time series that is a function of l for each bin k, the STFT then corresponds to a filter bank which decomposes the input signal into frequency channels or subbands. These two interpretations of the STFT are depicted with respect to the time–frequency plane in Fig. 12.1. Such time–frequency *tilings* have become common in the modern signal processing literature as a way to depict the time and frequency resolution of signal representations [12.4]. In the remainder of this Section, the *time-localized spectra* interpretation of the tiling is considered; the *filter bank* interpretation is treated further in Sects. 12.1.2 and 12.1.5.

The STFT is an *analysis* operation; it provides a representation of the signal that exposes some fundamental information about the signal. As will be discussed in Sect. 12.3, the STFT domain furthermore enables useful modifications of the signal, some of which are guided by this information captured in the STFT representation. In such scenarios, the STFT of the input time-domain signal is modified so as to create a desired effect in a new time-domain signal. To generate this modified time-domain signal, though, an appropriate *synthesis* operation is needed. Ideally, such a synthesis operation would reconstruct the original signal perfectly in the event that no STFT-domain modification is carried out. In the following, a synthesis procedure based on this *perfect reconstruction* property is developed.

For the sliding-window case where the STFT is interpreted as a series of time-localized spectra, it is straightforward to envision a reconstruction framework that is basically the opposite of the analysis: first, an inverse DFT (IDFT) of each local spectrum is carried out; then, the resulting signal frames are aggregated to synthesize the signal. If the DFT size is sufficiently large ($K \geq N$), the IDFT simply returns the windowed signal segment:

$$\begin{aligned}\hat{x}_l[n] &= \mathrm{IDFT}\{X[k,l]\} \\ &= w[n]x[n+lL], \quad 0 \leq n \leq N-1 \end{aligned} \tag{12.6}$$

or, with respect to a global time $t = n + lL$ instead of the in-frame index n,

$$\hat{x}_l[t-lL] = w[t-lL]x[t], \quad lL \leq t \leq lL+N-1\,, \tag{12.7}$$

where the latter expression is introduced to simplify the upcoming formulation. Regarding the size of the DFT, when $K > N$ the DFT is *oversampled*; this frequency-domain oversampling results in time-limited interpolation of the spectrum, which is analogous to the bandlimited interpolation that is characteristic of time-domain oversampling [12.13]. In the *undersampled* case $K < N$, time-domain aliasing is introduced. To avoid this difficulty, the condition $K \geq N$ is imposed at this point.

If the DFT is large enough that no aliasing occurs, reconstruction can be simply carried out by an *overlap-add* (OLA) process, possibly with a synthesis window [12.10, 14, 15]. Denoting this synthesis window by $v[n]$, the OLA reconstruction is given by

$$\begin{aligned}\hat{x}[t] &= \sum_l v[t-lL]\hat{x}_l[t-lL] \\ &= \sum_l v[t-lL]w[t-lL]x[t]\,. \end{aligned} \tag{12.8}$$

To construct the full output signal $\hat{x}[t]$, each signal frame generated by the IDFT is weighted by the synthesis window and added to any neighboring windows which overlap in time. Since $x(t)$ is not a function of l, (12.8) can be rewritten as

$$\hat{x}[t] = x[t]\left(\sum_l w[t-lL]v[t-lL]\right), \tag{12.9}$$

so perfect reconstruction is achieved if the analysis and synthesis windows satisfy the constraint

$$\sum_l w[t-lL]v[t-lL] = 1\,. \tag{12.10}$$

This constraint is similar to but somewhat more general than the perfect reconstruction constraints given in [12.1, 10–12, 14, 15]. Note that throughout this section the analysis and synthesis windows, as well as the signal, will be assumed to be real-valued.

In cases where $v[n]$ is not explicitly specified, the equivalent synthesis window is a rectangular window covering the same time span as $w[n]$. Then, the constraint in (12.10) becomes

$$\sum_l w[t-lL] = 1 \,. \tag{12.11}$$

The construction of windows with this property has been explored in the literature; a variety of such *perfect reconstruction windows* have been proposed, for example rectangular and triangular windows and the Blackman–Harris family, which includes the familiar Hanning and Hamming windows [12.16, 17]. These are also referred to as windows with the *overlap-add property* and will be denoted by $w_{\mathrm{PR}}[n]$ in the following derivations. Several issues are worth noting: *any* window function satisfies the condition in (12.11) when $L = 1$; for $L = N$, the only window that has the overlap-add property is a rectangular window of length N; for $L > N$, the overlap-add property cannot be satisfied as there are time gaps between successive frames.

There are a number of methods to design analysis and synthesis windows that satisfy (12.10). The methods discussed here rely on the use of familiar windows that satisfy (12.11) to jointly construct analysis and synthesis windows that satisfy (12.10); various analysis–synthesis window pairs designed in this way exhibit computational and modeling advantages. In one design approach, complementary powers of a perfect reconstruction window provide the analysis and synthesis windows:

$$\sum_l \left(w_{\mathrm{PR}}[t-lL]\right)^c \left(w_{\mathrm{PR}}[t-lL]\right)^{1-c} = 1$$

$$\Longrightarrow \begin{cases} \text{Analysis} & w[n] = \left(w_{\mathrm{PR}}[n]\right)^c \\ \text{Synthesis} & v[n] = \left(w_{\mathrm{PR}}[n]\right)^{1-c} \end{cases} . \tag{12.12}$$

The case $c = 1/2$, where the analysis and synthesis windows are equivalent, is of some interest because of its symmetry. A second approach is as follows; given a perfect reconstruction window $w_{\mathrm{PR}}[n]$ and an arbitrary window $b[n]$ that is strictly nonzero over the time support of $w_{\mathrm{PR}}[n]$, the overlap-add property can be rephrased as:

$$\sum_l w_{\mathrm{PR}}[t-lL]\left(\frac{b[t-lL]}{b[t-lL]}\right) = 1$$

$$\sum_l b[t-lL]\left(\frac{w_{\mathrm{PR}}[t-lL]}{b[t-lL]}\right) = 1$$

$$\Longrightarrow \begin{cases} \text{Analysis} & w[n] = b[n] \\ \text{Synthesis} & v[n] = \frac{w_{\mathrm{PR}}[n]}{b[n]} \end{cases} . \tag{12.13}$$

Windows of this form are used in the sinusoidal synthesis scheme described in [12.18, 19].

In practice, the STFT analysis window is generally chosen based on its frequency resolution and sidelobe behavior. Most commonly, the synthesis window is selected to provide perfect reconstruction given the analysis window; in cases where modifications of the STFT are carried out prior to synthesis, the synthesis window may be designed to minimize reconstruction artifacts [12.20].

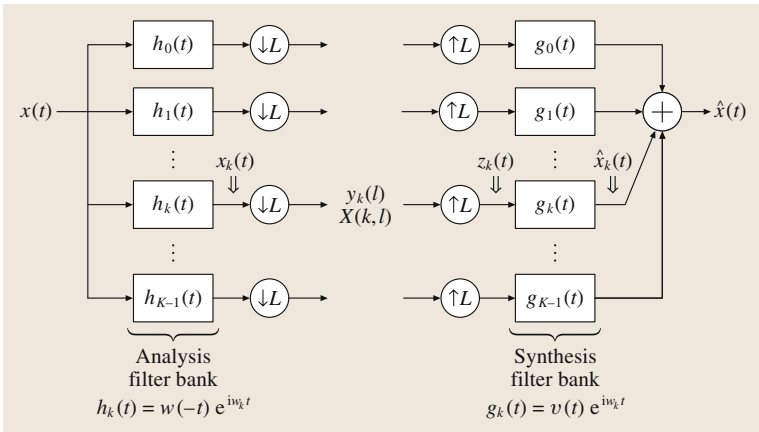


Fig. 12.2 Formulation of the short-time Fourier transform as a modulated filter bank

12.1.2 The STFT as a Modulated Filter Bank

Modulated filter banks, in which the constituent filters are modulated versions of a prototype low-pass filter, have received considerable attention in the signal processing literature, in large part due to the relative ease of designing such filter banks and their effectiveness for signal compression [12.4, 5, 21]. In this Section, we demonstrate that the STFT can be implemented with a modulated filter bank structure. Note that this equivalence of sliding-window sinusoidal transforms and modulated filter banks is indeed one of the basic concepts in the filter bank and subband coding literature [12.4].

Consider the general modulated filter bank shown in Fig. 12.2, with analysis filters $h_k[t]$ and synthesis filters $g_k[t]$. If the filters are defined as

$$h_k[t] = w[-t]\,\mathrm{e}^{\mathrm{i}\omega_k t}\,, \tag{12.14}$$

$$g_k[t] = v[t]\,\mathrm{e}^{\mathrm{i}\omega_k t}\,, \tag{12.15}$$

then the filter bank implements an STFT analysis–synthesis. The derivation of this correspondence is straightforward; using the notation in Fig. 12.2, where the subscript is now a subband index instead of a time index as earlier, the outputs of the analysis filter bank can be written

$$\begin{aligned} x_k[t] &= \sum_n h_k[n]x[t-n] \\ &= \sum_n w[-n]x[t-n]\,\mathrm{e}^{\mathrm{i}\omega_k n} \\ &= \sum_n w[n]x[n+t]\,\mathrm{e}^{-\mathrm{i}\omega_k n} \\ &= X[k,t]\,. \end{aligned} \tag{12.16}$$

Subsampling by L then yields the subband signals

$$\begin{aligned} y_k[l] &= x_k[lL] \\ &= \sum_n w[n]x[n+lL]\,\mathrm{e}^{-\mathrm{i}\omega_k n} \\ &= X[k,l]\,, \end{aligned} \tag{12.17}$$

which shows that the modulated analysis filter bank does indeed carry out the STFT analysis operation defined in (12.4). Note that subsampling by L in the filter bank subbands corresponds to using a hop size of L in the sliding-window framework.

In the synthesis filter bank, the subband signals $y_k[l]$ are first upsampled by L; the resulting signals can be expressed as

$$z_k[t] = x_k[t]\sum_l \delta[t-lL]\,, \tag{12.18}$$

which describes the effect of successive downsampling and upsampling on the signal $x_k[t]$. The outputs of the subband synthesis filters $g_k[t]$ can then be written

$$\begin{aligned} \hat{x}_k[t] &= z_k[t] * g_k[t] = \sum_n g_k[n]z_k[t-n] \\ &= \sum_n v[n]\,\mathrm{e}^{\mathrm{i}\omega_k n}x_k[t-n]\sum_l \delta[t-n-lL]\,. \end{aligned} \tag{12.19}$$

Using (12.16) then yields

$$\begin{aligned} \hat{x}_k[t] = \sum_n \sum_l \sum_m & v[n]w[m]x[t-n+m] \\ & \times \mathrm{e}^{\mathrm{i}\omega_k(n-m)}\delta[t-n-lL]\,. \end{aligned} \tag{12.20}$$

Perfect reconstruction is achieved if the outputs of the synthesis filters sum to the original signal:

$$\hat{x}[t] = \sum_k \hat{x}_k[t] = x[t]\,. \tag{12.21}$$

If the expression in (12.20) is substituted for $\hat{x}_k[t]$, the summation can be rearranged such that the sum over the frequency k is the innermost sum; since $\omega_k = 2\pi k/K$, this sum can be written as

$$\sum_{k=0}^{K-1} \mathrm{e}^{\mathrm{i}\omega_k(n-m)} = K\sum_r \delta[n-m+rK]\,. \tag{12.22}$$

If $|n-m| < K$ for all possible combinations of n and m, then the only relevant term in the right-hand sum is for $r = 0$, in which case the equation simplifies to

$$\sum_{k=0}^{K-1} \mathrm{e}^{\mathrm{i}\omega_k(n-m)} = K\delta[n-m]\,. \tag{12.23}$$

This restriction on the values of n and m corresponds to the $K \geq N$ constraint discussed in Sect. 12.1.1 for the sliding-window analysis–synthesis; namely, time-domain aliasing is introduced if n and m do not meet this criterion. It is thus assumed here, as in the earlier discussion, that $K \geq N$.

Using the result in (12.23), the output of the synthesis filter bank can be written

$$\begin{aligned} \hat{x}[t] = K\sum_n \sum_l & v[n]\delta[t-n-lL] \\ & \times \sum_m w[m]x[t-n+m]\delta[n-m]\,. \end{aligned} \tag{12.24}$$

The innermost sum is nonzero only for $n = m$, so the expression for $\hat{x}[t]$ can be simplified to

$$\hat{x}[t] = Kx[t]\sum_n \sum_l w[n]v[n]\delta[t-n-lL]\,. \tag{12.25}$$

Then, since $\delta[t-n-lL]$ is nonzero only for $n=t-lL$, this can be further simplified to

$$\hat{x}[t] = Kx[t] \sum_l w[t-lL]v[t-lL] \,. \tag{12.26}$$

Perfect reconstruction for the analysis–synthesis filter bank is thus achieved if

$$K \sum_l w[t-lL]v[t-lL] = 1 \,. \tag{12.27}$$

With the exception of the gain term, this is the same condition derived for the sliding-window framework. Because of this functional equivalence, the analysis–synthesis window pairs described in Sect. 12.1.1 can be used as prototype functions for perfect reconstruction modulated filter banks.

Note that if $L > 1$, the synthesis filter bank interpolates the subband signals. In the non-subsampled case ($L = 1$), when no interpolation is needed, perfect reconstruction can be achieved with any analysis–synthesis window pair for which $\sum_n w[n]v[n] \neq 0$. For example, the synthesis can be performed with the trivial filter bank $g_k[t] = \delta[t]$ if the analysis window satisfies the constraint

$$\sum_i w[n-i] = 1 \,, \tag{12.28}$$

which indeed holds for *any* window, within a gain term. The generality of this constraint is an example of the design flexibility that results from using oversampled or overcomplete approaches [12.4, 22–24].

12.1.3 Original Formulation of the STFT

The definition of the STFT given in (12.4) differs from that in the early literature [12.9–12, 14, 15], where the transform is given as

$$\tilde{X}[k,t] = \sum_{m=-\infty}^{\infty} \tilde{w}[t-m]x[m]\mathrm{e}^{-\mathrm{i}\omega_k m} \tag{12.29}$$

$$= \sum_{m=t}^{t+N-1} \tilde{w}[t-m]x[m]\mathrm{e}^{-\mathrm{i}\omega_k m} \,, \tag{12.30}$$

or in subsampled form as

$$\tilde{X}[k,l] = \sum_{m=lL}^{lL+N-1} \tilde{w}[lL-m]x[m]\mathrm{e}^{-\mathrm{i}\omega_k m} \,, \tag{12.31}$$

where $\tilde{w}[n]$ is again a time-localized window, as $w[n]$ in Sects. 12.1.1 and 12.1.2. The index m is a global time index, as opposed to the local time index n used in the earlier STFT formulation; the range of m in the sum, and hence the support of the window $\tilde{w}[n]$, is defined here in such a way that this alternate transform and $X[k,l]$ in (12.4) refer to the same N-point segment of the signal and can thus be compared. In some treatments, the STFT is expressed as in (12.31) but without the time reversal of the window [12.5]. This reversal of the time index affects the interpretation of the transform as a filter bank. More importantly, however, the filter bank interpretation is affected by the time reference of the expansion functions, as discussed below.

12.1.4 The Time Reference of the STFT

In the STFT formulation in (12.29–12.31), the signal is analyzed with respect to sinusoids whose time reference is absolute rather than relative to the windowed segments; for different segments, these functions have the same time reference, namely the time origin of the input signal $x[t]$. On the other hand, in (12.4), the time origin of the STFT sinusoids is the starting point of the signal segment in question; the phases of the STFT for a particular segment then refer to the time start of that segment. Note that the STFT can also be formulated such that the phase is referenced to the center of the time window, which is desirable in some cases [12.19, 25]; this extension plays a role in sinusoidal modeling, but such phase-centering will not be considered in the treatment of the STFT because of the slight complications it introduces.

The two formulations of the STFT have different ramifications for the interpretation of the STFT and the signal representation it yields; the differences can be seen by relating the two definitions. A change of index ($n = m - t$) in (12.30) yields

$$\begin{aligned}\tilde{X}[k,t] &= \sum_{n=0}^{N-1} \tilde{w}[-n]x[t+n]\mathrm{e}^{-\mathrm{i}\omega_k(t+n)} \\ &= \mathrm{e}^{-\mathrm{i}\omega_k t} \sum_{n=0}^{N-1} \tilde{w}[-n]x[t+n]\mathrm{e}^{-\mathrm{i}\omega_k n} \,.\end{aligned} \tag{12.32}$$

Letting $\tilde{w}[n] = w[-n]$,

$$\tilde{X}[k,t] = \mathrm{e}^{-\mathrm{i}\omega_k t} \sum_{n=0}^{N-1} w[n]x[t+n]\mathrm{e}^{-\mathrm{i}\omega_k n} \,. \tag{12.33}$$

Comparing this to (12.5), two simple relationships are clear:

$$X[k,t] = \tilde{X}[k,t]\,\mathrm{e}^{\mathrm{i}\omega_k t}\,, \tag{12.34}$$

$$|X[k,t]| = |\tilde{X}[k,t]|\,. \tag{12.35}$$

Note that these also hold in the subsampled case. The first relationship affects the interpretation of the STFT as a filter bank; the time signal $X[k,t]$ is a modulated version of the baseband envelope $\tilde{X}[k,t]$, so the corresponding filter banks for the two cases have different structures. The second expression affects the interpretation of the STFT as a series of time-localized spectra. The short-time magnitude spectra are equivalent in the two formulations. With respect to phase, however, the approaches differ in that $X[k,t]$ provides a *local* phase. An estimate of the local phase of each partial is important for building a sinusoidal signal model, so $X[k,t]$ is more useful for sinusoidal modeling than $\tilde{X}[k,t]$. This advantage will become more apparent in the discussion of sinusoidal modeling in Sect. 12.2.

12.1.5 The STFT as a Heterodyne Filter Bank

In [12.9–12, 14, 15], where the STFT is defined as in (12.31) with an absolute time reference, the transform can be interpreted as a filter bank with a heterodyne structure. Making the substitution

$$x_k[m] = x[m]\,\mathrm{e}^{-\mathrm{i}\omega_k m} \tag{12.36}$$

in (12.31) yields an expression that is immediately recognizable as a convolution

$$\tilde{X}[k,t] = \sum_{m=t}^{t+N-1} \tilde{w}[t-m]x_k[m]. \tag{12.37}$$

The filter $\tilde{w}[n]$ is typically low pass; it thus extracts the baseband spectrum of $x_k[m]$. According to the modulation relationship in (12.36), $x_k[m]$ is a version of $x[m]$ that has been modulated down by ω_k; thus, the baseband spectrum of $x_k[m]$ corresponds to the spectrum of $x[m]$ in the neighborhood of frequency ω_k. The k-th branch of the STFT filter bank thus extracts information about the signal in a frequency band around $\omega_k = 2\pi k/K$.

Figure 12.3 depicts one branch of a heterodyne STFT filter bank and provides an equivalent structure based on modulated filters [12.5]. Mathematically, the equivalence is straightforward:

$$\begin{aligned}\tilde{X}_1[k,t] &= \sum_m \tilde{w}[t-m](x[m]\,\mathrm{e}^{-\mathrm{i}\omega_k m}) \\ &= \mathrm{e}^{-\mathrm{i}\omega_k t}\sum_m (\tilde{w}[t-m]\,\mathrm{e}^{\mathrm{i}\omega_k(t-m)})x[m] \\ &= \tilde{X}_2[k,t]\,.\end{aligned} \tag{12.38}$$

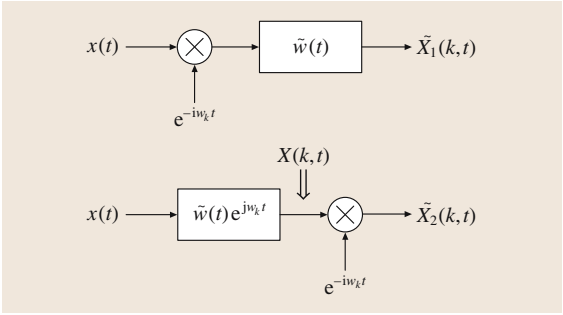


Fig. 12.3 One channel of a heterodyne filter bank for computing the STFT $\tilde{X}[k,t]$ defined in (12.29). The two structures are equivalent, as indicated in (12.38). The STFT $X[k,t]$ as defined in (12.5) is an intermediate signal in the second structure

Given the relationship in (12.34), namely that $X[k,t] = \tilde{X}[k,t]\,\mathrm{e}^{\mathrm{i}\omega_k t}$, it is clear that $X[k,t]$ is the immediate output of the modulated filter $\tilde{w}[n]\,\mathrm{e}^{\mathrm{i}\omega_k n}$ without the ensuing modulation to the baseband, as indicated in Fig. 12.3. This relationship sheds light on the impact of the time-base difference of the STFT variations in (12.4) and (12.31). Using an absolute time reference for the analysis leads to a heterodyne filter bank, while using a relative time reference results in a modulated filter bank; as shown above, the difference between the filter bank structures is a modulation of the subband signals to baseband.

12.1.6 Reconstruction Methods and Signal Models

In [12.1, 12] and other early treatments of the short-time Fourier transform, the two interpretations of the STFT lead to different reconstruction algorithms. For the sliding-window framework, where the STFT is viewed as a series of time-localized spectra, an OLA synthesis is formulated; for the filter bank case, the signal reconstruction approach is referred to as *filter bank summation* (FBS). In these early frameworks, the different reconstruction algorithms lead to different perfect reconstruction conditions that are related by a duality [12.12]. Here, with the STFT redefined to use a local time reference, the perfect reconstruction conditions for the two approaches are equivalent as derived in Sects. 12.1.1 and 12.1.2; that these conditions are ultimately equivalent is not particularly surprising since the representation of the STFT as a time–frequency tiling as in Fig. 12.1 suggests that a distinction between the two interpretations is actually artificial.

The reconstruction formula in an analysis–synthesis system corresponds to a *signal model*; with this in mind, it is worthwhile to consider the filter bank summation approach as it highlights the signal modeling inherent in the STFT. In the non-subsampled case, the FBS reconstruction formula for the original STFT of (12.29) is given by

$$\hat{x}[t] = \sum_k \tilde{X}[k,t]\,\mathrm{e}^{\mathrm{i}\omega_k t}\,. \tag{12.39}$$

For a fixed k, the STFT $\tilde{X}[k,t]$ can be interpreted as the amplitude envelope of a sinusoid with frequency ω_k; reconstruction is achieved by modulating each of the envelopes to the appropriate frequency and summing the resulting signals. The original signal is thus being modeled as a sum of amplitude-modulated sinusoids. For the phase-localized formulation of the STFT in (12.5), the FBS reconstruction formula is

$$\hat{x}[t] = \sum_k X[k,t]\,. \tag{12.40}$$

Here, $X[k,t]$ corresponds directly to a sinusoid at frequency ω_k rather than its amplitude envelope; the appropriate modulation is essentially built into the STFT representation.

12.1.7 Examples

Here the short-time Fourier transforms of several example signals are considered. This discussion should serve to clarify the behavior of the STFT and underscore some of its strengths and weaknesses as an analysis–synthesis tool.

Consider a signal consisting of a single sinusoid of amplitude A and frequency $\omega_0 = 2\pi f_0/F_\mathrm{s}$:

$$\begin{aligned} x_0[t] &= A\cos(\omega_0 t) \\ &= \frac{A}{2}\mathrm{e}^{\mathrm{i}\omega_0 t} + \frac{A}{2}\mathrm{e}^{-\mathrm{i}\omega_0 t}\,. \end{aligned} \tag{12.41}$$

Using (12.2), the STFT of $x_0[t]$ is given by

$$\begin{aligned} X_0[k,l] = \frac{A}{2}\mathrm{e}^{\mathrm{i}\omega_0 lL}\sum_{n=0}^{N-1} w[n]\,\mathrm{e}^{-\mathrm{i}(\omega_k-\omega_0)n} \\ + \frac{A}{2}\mathrm{e}^{-\mathrm{i}\omega_0 lL}\sum_{n=0}^{N-1} w[n]\,\mathrm{e}^{-\mathrm{i}(\omega_k+\omega_0)n}\,. \end{aligned} \tag{12.42}$$

Denoting the discrete-time Fourier transform (DTFT) of $w[n]$ by $W(\omega)$, the STFT can be rewritten as

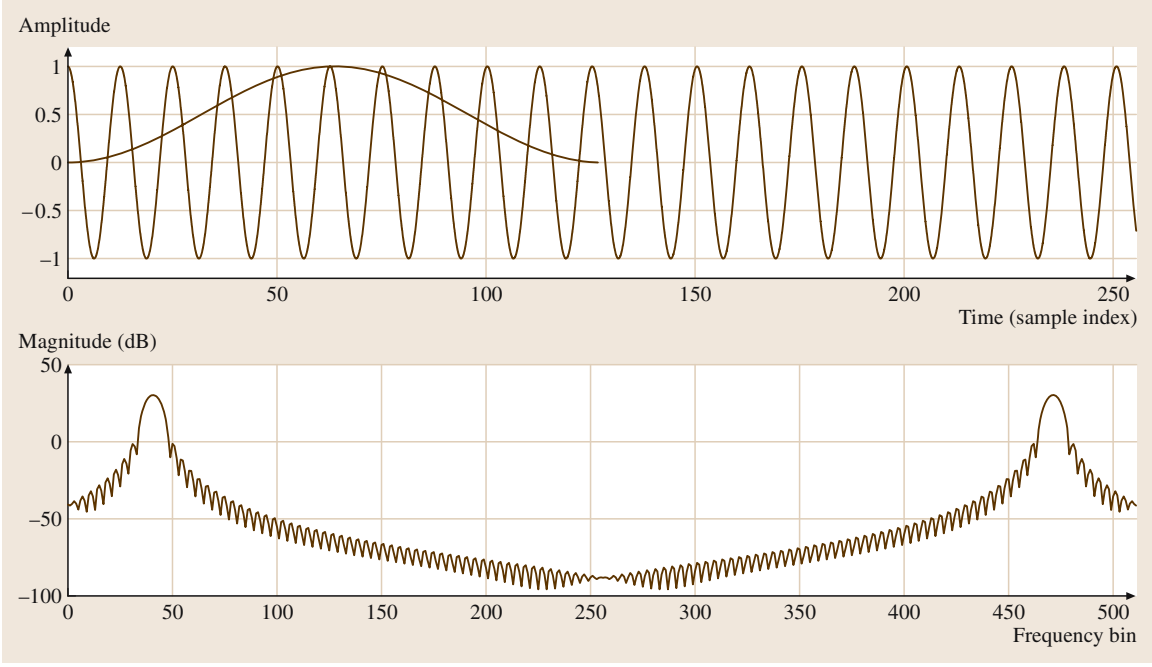


Fig. 12.4 A sinusoid and its STFT magnitude for one time window. The spectrum exhibits two main lobes as indicated by (12.43)

$$X_0[k,l] = \frac{A}{2} e^{i\omega_0 lL} W(\omega_k - \omega_0) + \frac{A}{2} e^{-i\omega_0 lL} W(\omega_k + \omega_0) . \quad (12.43)$$

The STFT of a single real sinusoid is thus a sum of two modulated versions of the window spectrum $W(\omega)$, each with a different phase based on the frequency ω_0, the time index l, and the hop size L. In a given frame of $X_0[k,l]$, then, two peaks should be apparent in the magnitude of the time-localized spectrum, one at $+\omega_0$ and one at $-\omega_0$; this is illustrated in Fig. 12.4 for the case $\omega_0 = 0.5$, $N = 128$, and $K = 512$. Note that *negative* frequencies are shown in the plots as bin indices $K/2 < k < K$. The two modulated copies of $W(\omega)$ interfere with each other such that the peaks in $|X_0[k,l]|$ are not necessarily at $\pm\omega_0$. If the frequency ω_0 is small enough that the peaks in the modulated copies actually overlap, the resulting deviation in the peak location can be substantial; for sufficiently high ω_0 such that the interference is limited to low-level sidelobes, as in Fig. 12.4, the error may be negligible. As will be seen in Sect. 12.2, the spectral peaks in the STFT play a key role in the analysis methods for sinusoidal modeling, as does the interference between multiple peaks.

For the constant-frequency sinusoid $x_0[t]$, the STFT varies only slightly from frame to frame, as indicated in (12.43). In contrast, consider the chirp signal

$$x_1[t] = A\cos(\omega_0 t + \alpha_0 t^2) . \quad (12.44)$$

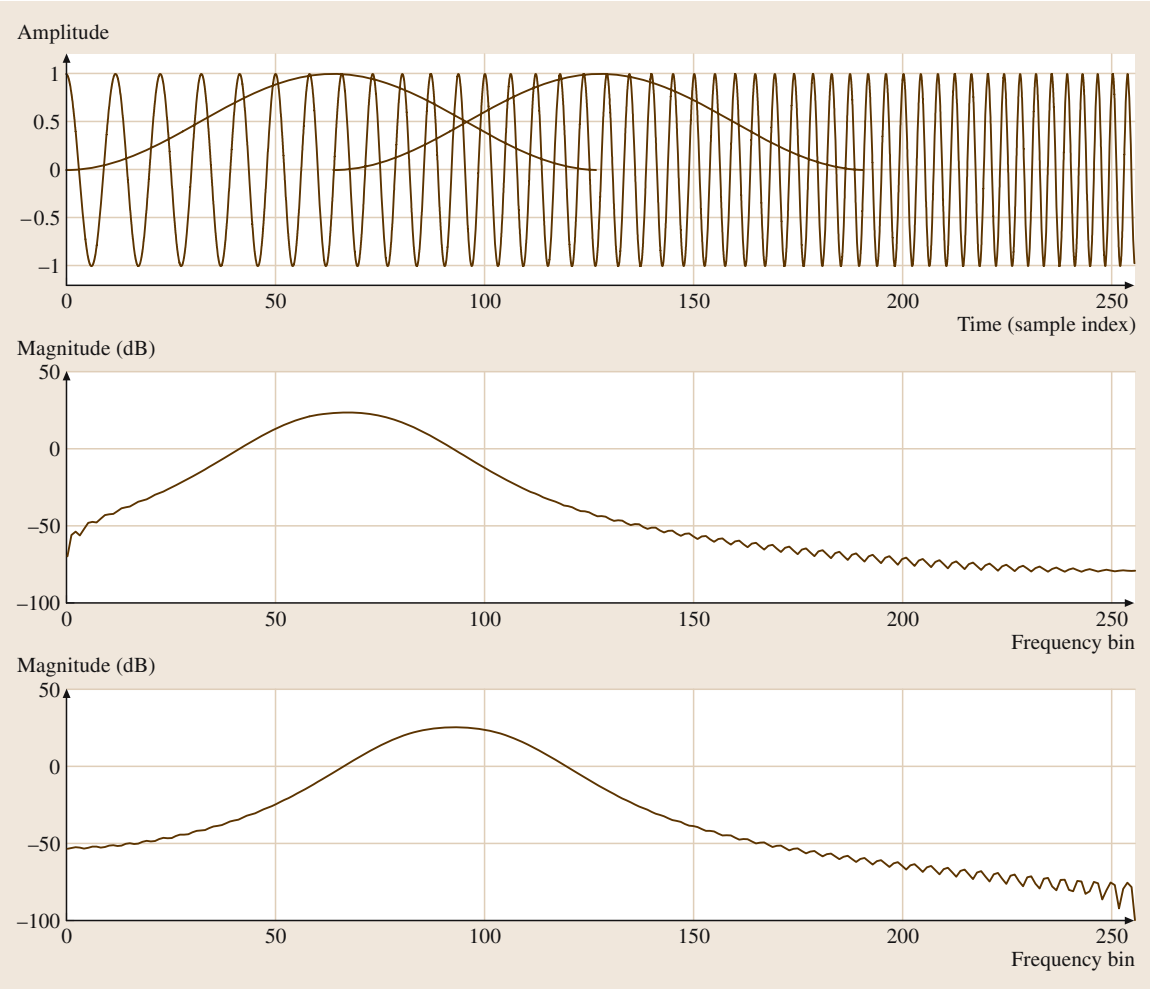


Fig. 12.5 A chirp and the magnitude of its STFT for the two time windows indicated in the top plot

When the signal characteristics vary from frame to frame as for $x_1[t]$, the STFT in turn varies from frame to frame; from a filter bank perspective, the chirp moves across the subbands as time progresses. The frame-to-frame variation of the STFT is shown in Fig. 12.5 for $\omega_0 = 0.5$ and $\alpha_0 = 0.0025$; note that only the positive-frequency bins $0 \le k \le \frac{K}{2}$ are included in the plot. The progression of the chirp across the subbands is illustrated in Fig. 12.6, which depicts the (non-subsampled) STFT filter bank subband signals for several subbands.

As a final synthetic example, consider a sum of weighted sinusoids

$$x_2[t] = \sum_q A_q \cos(\omega_q t + \phi_q) \,, \tag{12.45}$$

where an arbitrary phase ϕ_q has been added to each component for the sake of generality. By linearity, the STFT of $x_2[t]$ can easily be derived from (12.43) as

$$X_2[k,l] = \sum_q \frac{A_q}{2} \mathrm{e}^{\mathrm{i}(\omega_q lL + \phi_q)} W(\omega_k - \omega_q) + \frac{A_q}{2} \mathrm{e}^{-\mathrm{i}(\omega_q lL + \phi_q)} W(\omega_k + \omega_q) \,. \tag{12.46}$$

In this compound signal, there is interference not only between positive- and negative-frequency components (as for $x_0[t]$), but also between components of different frequencies. A multi-sinusoid signal and its STFT magnitude are depicted in Fig. 12.7; note that the magnitude differs somewhat from window to window due to the time-dependent interaction of the various components.

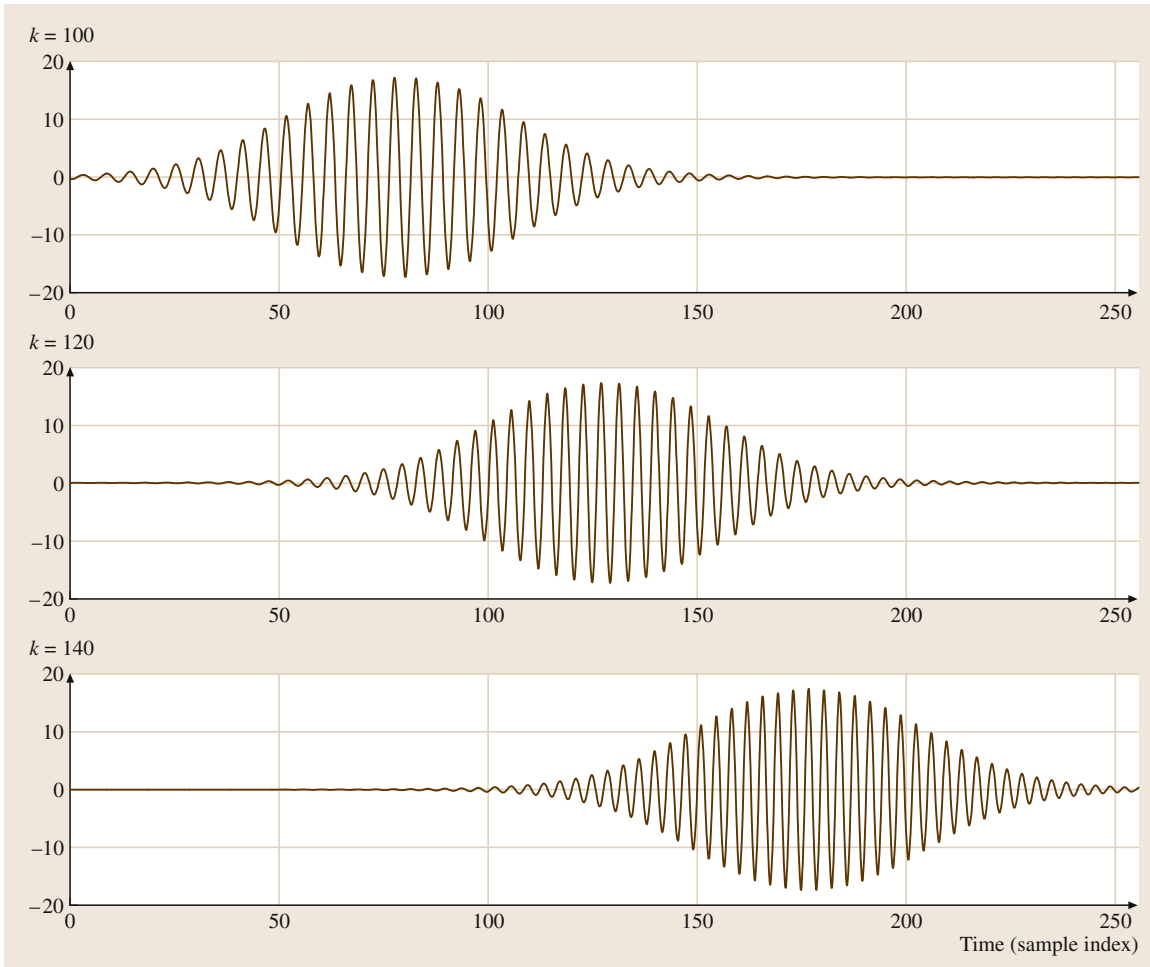


Fig. 12.6 Non-subsampled STFT subband signals (real part) for the chirp signal of Fig. 12.5

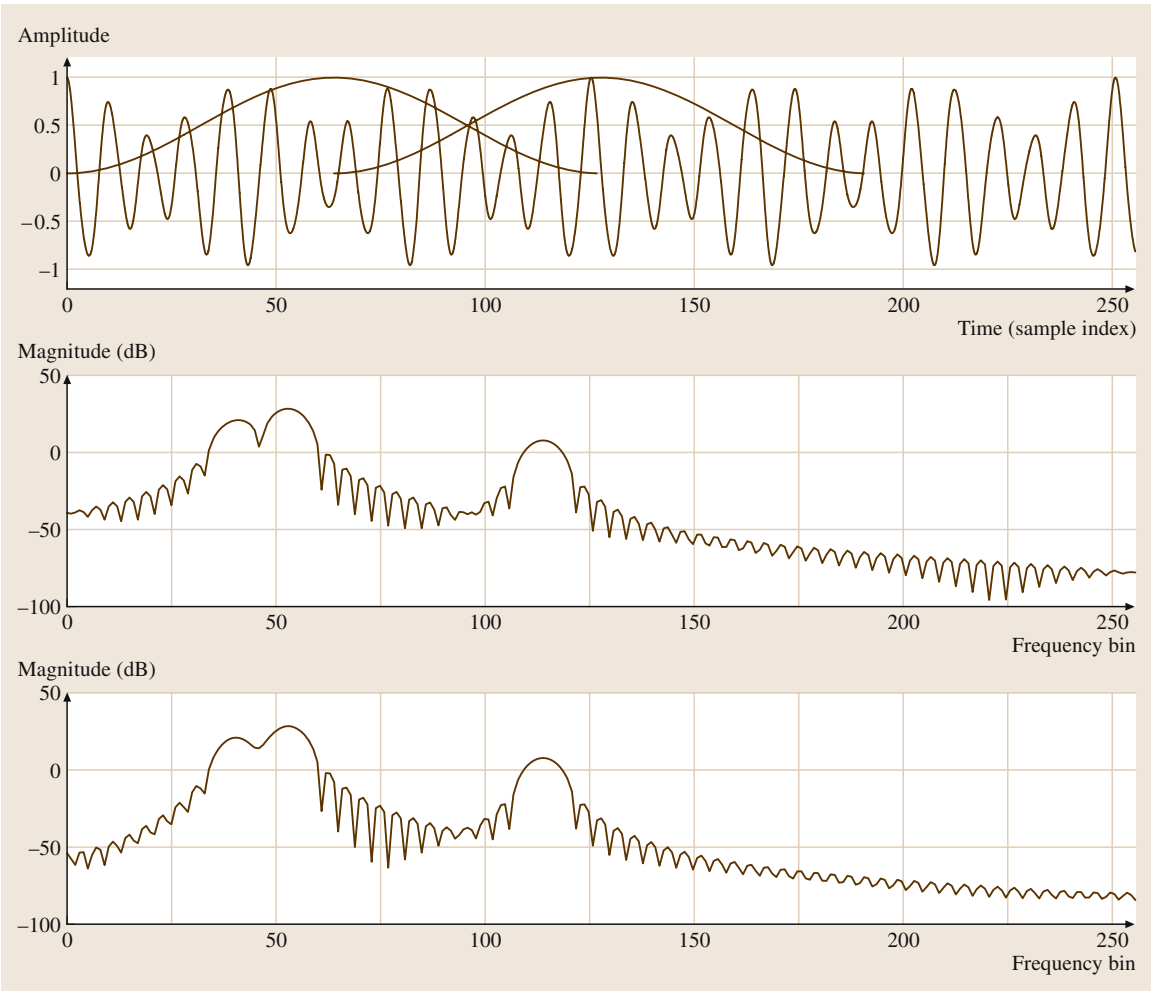


Fig. 12.7 A sum-of-sinusoids signal and the magnitude of its STFT for the two indicated time windows; the spectrum exhibits lobes centered at the frequencies of the constituent sinusoids, which may overlap for closely spaced frequencies

These example signals are intended as a starting point for understanding the practical behavior of the STFT. Signals of interest for speech processing applications are of course typically more complex than sinusoids or chirps. However, for many speech signals, such as the example in Fig. 12.8, a sum of weighted sinusoids is a reasonable model. In a given frame, the STFT magnitude exhibits spectral peaks that can be interpreted as local sinusoids. Also, it is clear that the STFT magnitude changes from frame to frame, but some of the spectral peaks persist; these can be interpreted as persistent sinusoids in the speech signal. These observations suggest that the signal could be approximated as a sum of short-term or potentially long-term sinusoids, as will be explored in Sect. 12.2. Figure 12.9 depicts the (non-subsampled) STFT filter bank subband signals for several subbands for the speech signal of Fig. 12.8; the subband signals correspond to amplitude-modulated sinusoids at the subband center frequencies. Recall from (12.40) that the original speech signal can be reconstructed directly as the sum of the non-subsampled STFT subband signals; this example sheds light on the interpretation of this FBS reconstruction as a high-order sinusoidal model of the signal (with one sinusoid per STFT subband). This interpretation serves as another motivation for the sinusoidal modeling approaches to be described in Sect. 12.2 in that the sinusoidal model attempts to extract a representa-

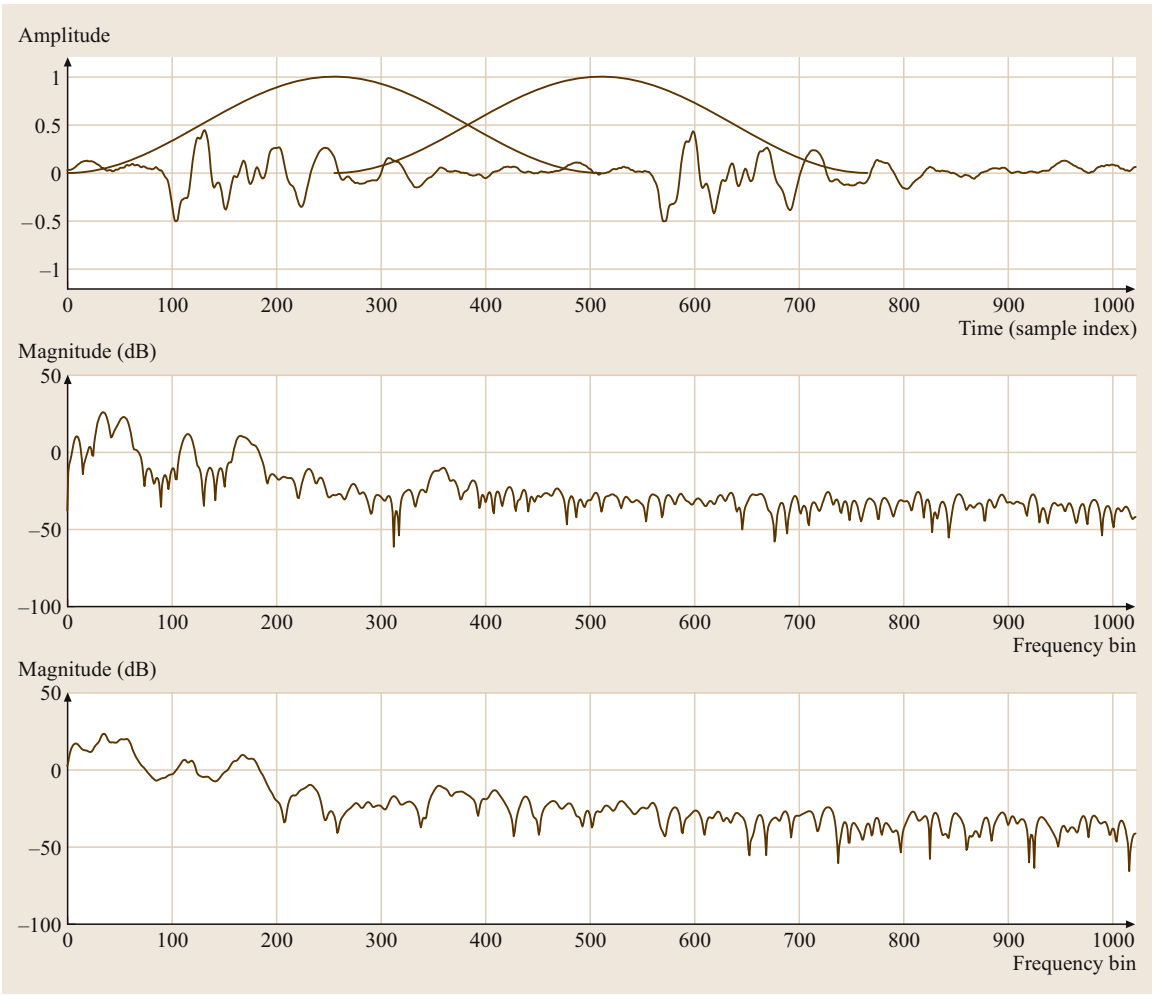


Fig. 12.8 A speech signal and the magnitude of its STFT for the two indicated time windows; the STFT clearly changes from frame to frame, but some spectral peaks persist

tion that consists of fewer sinusoids and is therefore more amenable to some processing applications (such as coding).

12.1.8 Limitations of the STFT

The short-time Fourier transform has proven effective for many applications in speech signal processing, some of which will be discussed later in Sect. 12.3. In this section, several limitations of the STFT are described; these undermine the applicability of the STFT for some problems, and indeed served as the motivation for further advances in filter bank and signal modeling methods.

Signal Coding

Filter banks and sinusoidal transforms are widely used for signal coding since they can be designed to expose the salient signal structure and thereby enable robust perceptually motivated quantization schemes [12.26]. In the signal coding scenario, three design conditions arise for the signal transform or representation: it should provide perfect (or at least near-perfect) reconstruction in the event that no quantization is imposed on the transform coefficients; it should be critically sampled, i. e., it should not introduce additional data; and, the synthesis algorithm should use overlap-add to reduce blocking artifacts in the reconstruction. It was shown in Sect. 12.1.1 that the STFT can provide perfect recon-

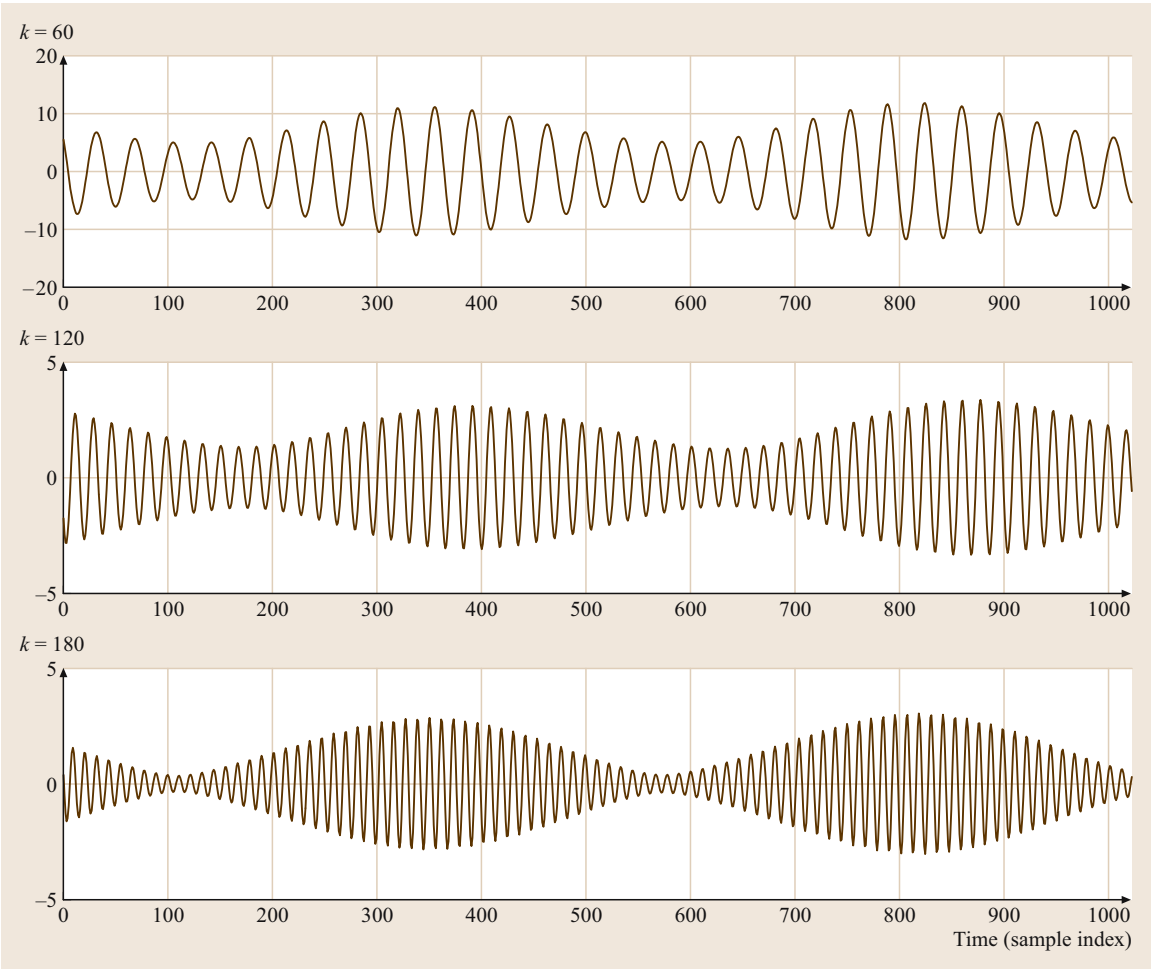


Fig. 12.9 Non-subsampled STFT subband signals (real part) for the speech signal of Fig. 12.8

struction with overlap-add synthesis; however, the rate of the STFT representation exceeds that of the original signal. This data increase is straightforward to quantify using the sliding-window framework. For a real-valued signal, each time-localized STFT spectrum comprises K unique values: one real value at $k = 0$, one real value at $k = K/2$, and $K/2 - 1$ complex values for $0 < k < K/2$. To achieve perfect reconstruction of a frame, $K \geq N$ is required to avoid time-domain aliasing. The data rate is then K/L since K values are needed to represent each L samples perfectly. This can only take on a value of one, i. e., achieve critical sampling, when $K = L = N$, meaning that perfect reconstruction with critical sampling cannot be achieved with an STFT whose frames overlap.

To overcome this limitation of the STFT, sliding-window transforms that admit cancelable time-domain aliasing have been developed; these achieve perfect reconstruction at critical sampling in an overlap-add framework [12.21, 27, 28]. The STFT cannot be structured so as to incorporate such time-domain aliasing cancelation [12.4, 19]. Further discussion of this issue is beyond the scope of this chapter; the key point is that the STFT is not directly practical for signal coding due to an inherent data increase in the transform representation. As will be discussed in Sect. 12.2, this problem is addressed in the sinusoidal model by applying a parametric representation to the STFT to achieve compaction.

Time–Frequency Localization

As discussed in the previous section, the design of STFT filter banks is very limited in the critically sampled case. The only real-valued prototype windows that lead to

perfect reconstruction filter banks are rectangular windows; this result is a discrete-time equivalent of the Balian–Low theorem, which essentially states that there are no continuous-time orthogonal short-time Fourier transform bases that are well localized in both time and frequency [12.4]. Further discussion is beyond the scope of this chapter; the interested reader is referred to the literature on cosine-modulated filter banks [12.4,5], which can achieve good localization in time and frequency and which can also satisfy the design conditions given earlier for signal coding.

Signal Adaptivity

Perhaps the most obvious (or at least most easily demonstrable) limitation of the short-time Fourier transform results from its fixed structure. Since the STFT analysis–synthesis does not adapt to the signal behavior, it yields a *noncompact* model for nonstationary signals [12.19]. Consider again the chirp example from Fig. 12.6, which illustrated that a sinusoid with time-varying frequency moves across the subbands as the frequency evolves. The subbands of the STFT filter bank do not characterize the chirp signal as a single evolving sinusoid but instead as a conglomeration of short-lived components, which is obviously not a compact way to represent a simple chirp. A similarly dispersed representation is evident in Fig. 12.9 for a pseudoperiodic speech signal. This lack of compactness suggests a shortcoming of the STFT for low-rate signal coding; furthermore, it indicates that the STFT does not directly provide a high-level understanding of the signal behavior. These issues motivate the use of a signal-dependent sparse parametric model of the STFT to capture the most relevant information about the signal, be it for coding or analysis. Such parametrization is the key idea in the sinusoidal models to be considered in Sect. 12.2.

12.2 Sinusoidal Models

Sinusoidal modeling can be viewed as an evolution of short-time Fourier transform and phase vocoder techniques. In this treatment, the shortcomings of the STFT serve to motivate the general sinusoidal model, which is developed as a parametric extension of the STFT. Two sinusoidal modeling approaches are described, one in which the prominent magnitude peaks of the STFT are tracked from frame to frame and one in which each time-localized spectrum of the STFT is modeled as a sum of sinusoids. These are loosely analogous to the interpretations of the STFT discussed in Sect. 12.1, but the parametric analysis–synthesis ultimately extends the flexibility of the representation beyond that afforded by sliding-window transforms or filter banks. The latter method of modeling the time-localized spectra is treated in detail in Sects. 12.2.4 and 12.2.5, where a sinusoidal analysis–synthesis algorithm based on *matching pursuit* is presented.

12.2.1 Parametric Extension of the STFT

The term *vocoder*, a contraction of *voice* and *coder*, was coined by early speech analysis–synthesis researchers [12.29], but has now become a general designation for a range of algorithms which are by no means limited to voice coding applications. Of particular interest as a starting point for considering the parametric extension of the STFT is the *channel vocoder*, which originated as a voice coder that represented a speech signal based on the characteristics of filter bank channels or subbands. In the STFT context, the speech signal is filtered into a large number of channels using an STFT analysis filter bank. Each of the subbands is then modeled in terms of its short-time energy; with respect to the k-th channel, this provides an amplitude envelope $A_k[t]$ that is used to modulate a sinusoidal oscillator at the channel center frequency ω_k. The outputs of these oscillators are then accumulated to reconstruct the signal.

The channel vocoder parameterizes the subband signal in terms of its energy or amplitude only; the *phase vocoder* is an extension that also includes the phase behavior in the model parameterization [12.30]. In the literature, the term *phase vocoder* is sometimes used as a synonym for the short-time Fourier transform [12.31], but in some applications the approach involves interpreting the STFT analysis data with respect to a structure like the one shown in Fig. 12.10, where the subband signals are parameterized in terms of magnitude envelopes and functions that describe the frequency and phase evolution. These functions serve as inputs to a bank of oscillators that reconstruct the signal from the parametric model [12.10, 30, 32, 33]. If the analysis filter bank is subsampled, the sample-rate oscillator control functions are derived from the subsampled frame-rate STFT representation. This phase vocoder structure has

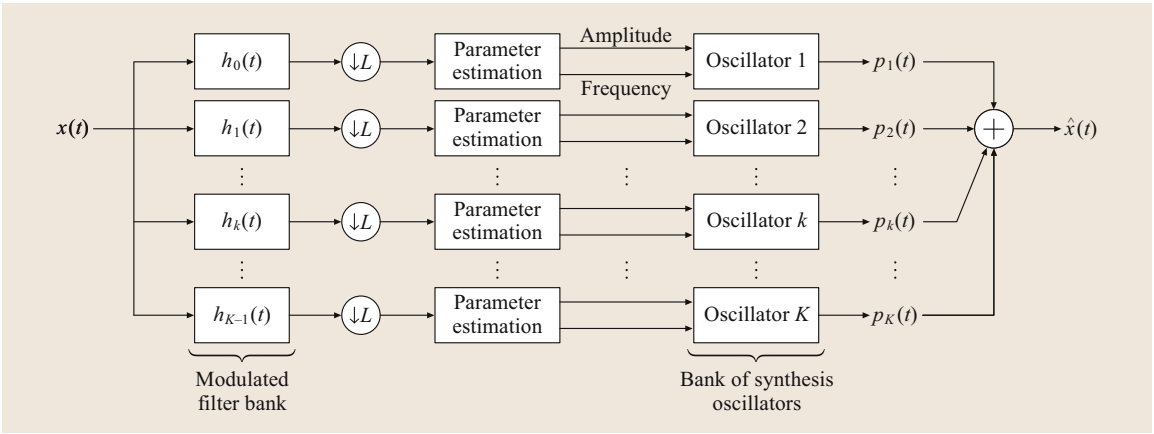


Fig. 12.10 Block diagram of the phase vocoder. The amplitude and frequency (total phase) control functions for the K oscillators are derived from the filter bank subband signals by the parameter estimation blocks

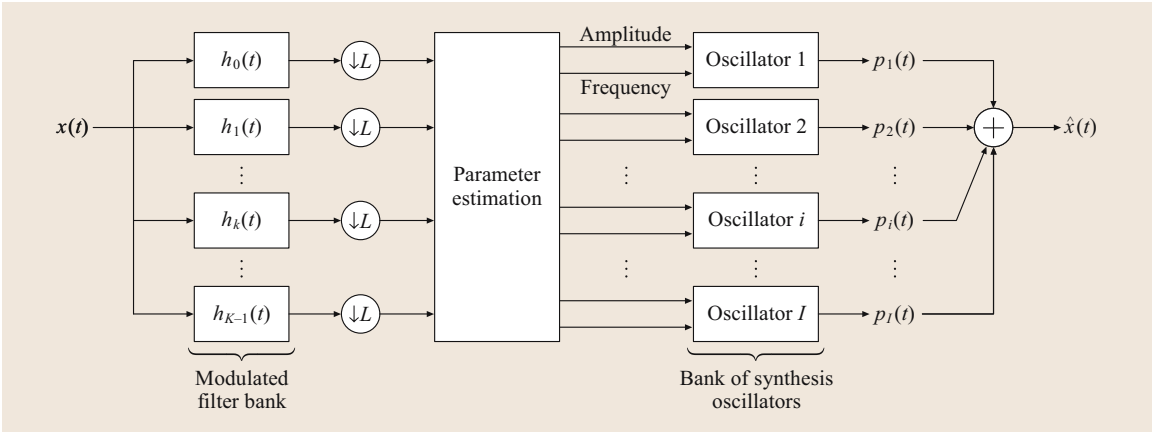


Fig. 12.11 Block diagram of the partial-tracking sinusoidal model. The amplitude and frequency (total phase) control functions are derived from the filter bank outputs by tracking spectral peaks in time as they move from band to band for an evolving signal. The parameter estimation block detects and tracks spectral peaks; unless I is externally constrained, the number of peaks detected dictates the number of oscillators used for synthesis

been widely applied to modification of speech and audio signals [12.31, 34–36].

The phase vocoder as depicted in Fig. 12.10 does not address the shortcoming of the STFT, i. e., lack of signal adaptivity. While its parametric nature does facilitate some modifications, the structure is still of limited use for compact modeling of evolving signals. A further generalization leads to a sinusoidal model in which partials, or sinusoidal components, are tracked in time from frame to frame, i. e., a *partial-tracking* sinusoidal model. The basic observation underpinning this model is that, if the signal consists of one nonstationary sinusoid such as a chirp, synthesis can be achieved with one oscillator. There is no need to implement an oscillator for every branch of the analysis filter bank. Instead, the outputs of the analysis bank can be examined across frequency for peaks, which are interpreted as sinusoids in the signal in the corresponding time frame. These spectral peaks can then be tracked from frame to frame as the signal evolves, and only one oscillator per tracked peak is required for synthesis. This structure is depicted in Fig. 12.11. Note that, because the synthesis for this case is based on a bank of sinusoidal oscillators, the prototype window for the analysis filter bank does not have to satisfy an overlap-add property.

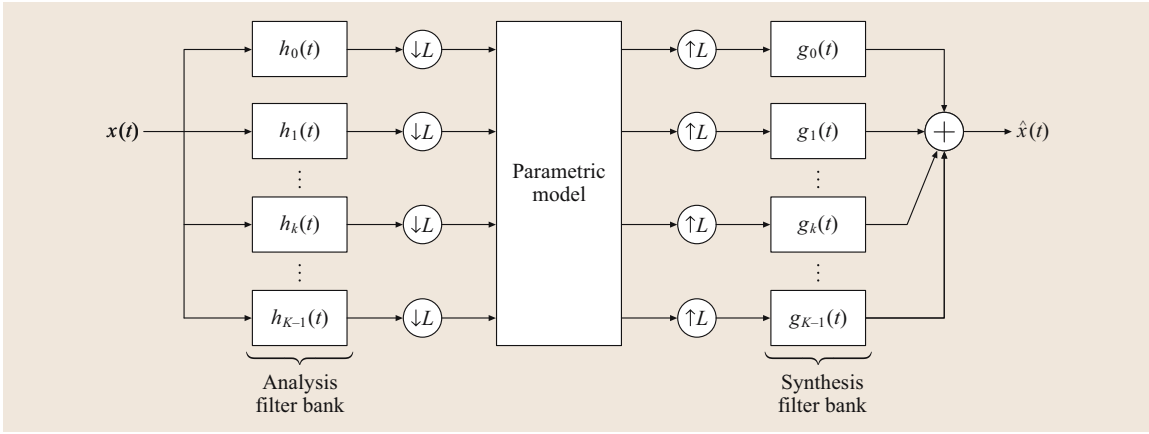


Fig. 12.12 Block diagram of frequency-domain synthesis for sinusoidal modeling in an overlap-add framework. The parametric model includes the sinusoidal analysis and the construction of short-time spectra from the analysis data. The IDFT/OLA process in the frequency-domain synthesizer can be interpreted as an STFT synthesis filter bank

As an alternative to the partial-tracking approach, each time-localized spectrum of the STFT can be approximated as a sum of short-term sinusoids. Synthesis is then carried out using the spectra of the sinusoidal approximations; each approximate spectrum is inverse-transformed and the full reconstruction is carried out using an overlap-add process. Such *overlap-add* sinusoidal modeling is depicted in Fig. 12.12, where the synthesis procedure is interpreted as a filter bank. In this approach, the overlap-add property of the analysis and synthesis windows plays an important role.

12.2.2 The Sinusoidal Signal Model

The basic idea in sinusoidal modeling is to approximate a signal as a sum of sinusoids whose parameters vary in time:

$$\hat{x}[t] = \sum_{i=1}^{I[t]} a_i[t] \cos \Phi_i[t], \qquad (12.47)$$

where $a_i[t]$ is the amplitude variation of the i-th sinusoidal partial; the total phase $\Phi_i[t]$ represents its frequency and phase variation. Note that the number of sinusoidal components $I[t]$ may also vary in time (depending on modeling error criterion and/or bit-rate constraints). The formulation is expressed in terms of the reconstructed signal $\hat{x}[t]$ since this is a *synthesis* model; the goal in the analysis, then, is to extract model parameters from the original signal $x[t]$ so as to create a reasonable approximation via the model-based reconstruction, i. e., such that $\hat{x}[t] \approx x[t]$.

12.2.3 Sinusoidal Analysis and Synthesis

Analysis for the sinusoidal model can be carried out in a variety of ways. The fundamental task is that of finding time-varying amplitude, frequency, and phase parameters for a set of sinusoids such that their sum accurately represents the behavior of the original signal. In some methods, the parameters are extracted from the signal simultaneously, i. e., the parameter sets of all $I[t]$ partials are determined concurrently; in other cases, parameters are extracted sequentially, i. e., one component at a time. Numerous approaches have been considered for the estimation of such sinusoidal parameters, either simultaneously or sequentially; this multiplicity of approaches is partly a result of the wide range of applications beyond speech and audio processing. In this chapter, we limit consideration to methods based on the short-time Fourier transform, which was fundamental to early sinusoidal models [12.37, 38]. In this Section, we discuss these methods briefly before considering a sequential analysis scheme in detail in Sects. 12.2.4 and 12.2.5.

In simultaneous sinusoidal analysis based on the STFT, the model parameters for the l-th time frame are determined from the short-time magnitude spectrum $|X[k, l]|$ via peak-picking, perhaps with a curve-fitting scheme employed to account for the spectral shape of the analysis window [12.19, 37, 38]. The I most prominent peaks are determined, and the model parameters $\{a_i, \omega_i, \phi_i\}_l$ are then given respectively by the magnitude, frequency, and phase of $X[k, l]$ at these spectral magnitude peaks. As illustrated in the STFT examples

of Sect. 12.1.7, such simultaneous estimation can be impacted by interference between multiple sinusoidal components in the signal.

In sequential analysis, the idea is to overcome the component interference problem by first selecting the most dominant component, then subtracting its contribution to the signal model from the analysis STFT, and thereafter iterating the estimation on the residual. This is referred to as analysis by synthesis [12.39]. By subtracting a component, the estimation of subsequent sinusoidal parameters is not affected by interference (spectral overlap) with that component. So, since the more-dominant components are extracted first, the estimation of minor components is not biased by their presence. Note of course that the parameter estimation is biased by the presence of components not yet extracted.

Either simultaneous or sequential analysis based on the STFT yields a set of sinusoidal parameters $\{a_i, \omega_i, \phi_i\}_l$ for each time frame of the STFT. There are two distinct methods to reconstruct an approximation of the original signal based on these parameter sets: parameter interpolation and overlap-add [12.19,37,39]. In parameter interpolation, partials are tracked from frame to frame via parameter matching, e.g., spectral peaks at the same or similar frequencies in successive time frames are assumed to correspond to a longer-term underlying sinusoidal component that spans multiple signal frames. The reconstruction is then achieved by using amplitude and phase interpolation to connect the matched parameter sets. In overlap-add modeling, the original signal is windowed and each windowed segment is modeled as a sum of sinusoids; for synthesis, each windowed segment is reconstructed and OLA is used to generate the full output. This approach overcomes the frame-to-frame tracking difficulties of time-domain interpolation and enables accurate approximation of the signal waveform, which is not readily achievable in interpolative synthesis.

Details on partial tracking for sinusoidal synthesis are available in the literature [12.37,38,40–42]; overlap-add synthesis for the sinusoidal model is discussed in [12.19,39,43,44]. In the remainder of Sect. 12.2, we focus on the overlap-add approach, providing a detailed treatment of overlap-add sinusoidal modeling based on the matching pursuit algorithm. The general pursuit algorithm is discussed in Sect. 12.2.4, and the specific sinusoidal case is developed in Sect. 12.2.5.

12.2.4 Signal Modeling by Matching Pursuit

Matching pursuit (MP) is an iterative analysis-by-synthesis algorithm for deriving compact signal approximations of the form

$$x[t] \approx \sum_{i=1}^{I} \alpha_i g_i[t], \tag{12.48}$$

where the expansion functions $g_i[t]$ are chosen from a dictionary [12.45]. MP operates by finding the element in the dictionary that best matches the signal, projecting the signal onto that element, and iterating on the projection residual. For a dictionary D, the task at the i-th stage of MP is to find the function $g_i[t] \in D$ and the coefficient α_i which minimize the norm of the residual

$$r_{i+1}[t] = r_i[t] - \alpha_i g_i[t], \tag{12.49}$$

where the initial condition is $r_1[t] = x[t]$. The solution is given by orthogonality [12.19,45]; for unit-norm dictionary functions $\boldsymbol{g}_i$,

$$\|\boldsymbol{r}_{i+1}\|^2 = \|\boldsymbol{r}_i\|^2 - |\alpha_i|^2 \tag{12.50}$$

$$\alpha_i = \langle \boldsymbol{g}_i, \boldsymbol{r}_i \rangle = \boldsymbol{g}_i^{\mathrm{H}} \boldsymbol{r}_i \tag{12.51}$$

$$\boldsymbol{g}_i = \arg\max_{\boldsymbol{g} \in D} |\langle \boldsymbol{g}, \boldsymbol{r}_i \rangle| , \tag{12.52}$$

where the signals are treated as column vectors and H denotes the conjugate transpose. The optimal function $\boldsymbol{g}_i$ is the one whose correlation with the signal has the largest magnitude.

Each iteration in MP requires all of the correlations between the dictionary functions and the current residual; these can be derived efficiently using an update formula [12.45]:

$$\langle \boldsymbol{g}, \boldsymbol{r}_{i+1} \rangle = \langle \boldsymbol{g}, \boldsymbol{r}_i \rangle - \alpha_i \langle \boldsymbol{g}, \boldsymbol{g}_i \rangle . \tag{12.53}$$

The dictionary cross-correlation terms $\langle \boldsymbol{g}, \boldsymbol{g}_i \rangle$ can generally be precomputed and stored to reduce the computation required for each iteration. For some dictionaries, the correlation can be carried out efficiently without using this update formulation [12.46].

In basic MP, each iteration searches for a single function for the signal model. An alternative is to consider subspaces at each iteration; the goal in such *subspace pursuit* is to find the matrix G that minimizes the norm of $\boldsymbol{r}_{i+1} = \boldsymbol{r}_i - G\boldsymbol{\alpha}$, where $\boldsymbol{\alpha}$ is a coefficient vector and the columns of G are dictionary functions [12.19,46]. The orthogonality constraint $\langle \boldsymbol{r}_i - G\boldsymbol{\alpha}, G \rangle = 0$ yields the solution:

$$||\boldsymbol{r}_{i+1}||^2 = ||\boldsymbol{r}_i||^2 - \boldsymbol{r}_i^{\mathrm{H}} G (G^{\mathrm{H}} G)^{-1} G^{\mathrm{H}} \boldsymbol{r}_i , \tag{12.54}$$

$$\alpha_i = (G^{\mathrm{H}} G)^{-1} G^{\mathrm{H}} \boldsymbol{r}_i , \tag{12.55}$$

$$G_i = \arg\max_{G \in D} \{\boldsymbol{r}_i^{\mathrm{H}} G \alpha_i\} . \tag{12.56}$$

The search in (12.56) is computationally intensive unless G consists of orthogonal vectors or has some other special structure [12.19, 47].

In some scenarios, such as sinusoidal modeling, it is useful to carry out pursuit based on the subspace spanned by a function and its complex conjugate. Here, the columns of G are a function $\boldsymbol{g}$ and its conjugate $\boldsymbol{g}^*$. If the signal $\boldsymbol{r}_i$ is real and if $\boldsymbol{g}$ and $\boldsymbol{g}^*$ are linearly independent, the subspace pursuit can be simplified. Using (12.55), the optimal coefficients for a conjugate pair $\{\boldsymbol{g}, \boldsymbol{g}^*\}$ are

$$\begin{pmatrix} \alpha_i \\ \alpha_i^* \end{pmatrix} = \frac{1}{1-|c|^2} \begin{pmatrix} \langle \boldsymbol{g}, \boldsymbol{r}_i \rangle - c \langle \boldsymbol{g}, \boldsymbol{r}_i \rangle^* \\ \langle \boldsymbol{g}, \boldsymbol{r}_i \rangle^* - c^* \langle \boldsymbol{g}, \boldsymbol{r}_i \rangle \end{pmatrix} , \quad (12.57)$$

where $c = \langle \boldsymbol{g}, \boldsymbol{g}^* \rangle$ and where α_i is again a scalar, and the pursuit optimization of (12.56) is given by

$$\boldsymbol{g}_i = \arg \max_{\boldsymbol{g} \in D} (\langle \boldsymbol{g}, \boldsymbol{r}_i \rangle^* \alpha_i + \langle \boldsymbol{g}, \boldsymbol{r}_i \rangle \alpha_i^*) \quad (12.58)$$

$$= \arg \max_{\boldsymbol{g} \in D} (\mathrm{Re}\{\langle \boldsymbol{g}, \boldsymbol{r}_i \rangle^* \alpha_i\}) . \quad (12.59)$$

The algorithm simply searches for the function $g_i[t]$ that minimizes the norm of the residual

$$\begin{aligned} r_{i+1}[t] &= r_i[t] - \alpha_i g_i[t] - \alpha_i^* g_i^*[t] \\ &= r_i[t] - 2\,\mathrm{Re}\{\alpha_i g_i[t]\} . \end{aligned} \quad (12.60)$$

The resulting signal approximation has the form

$$x[t] \approx 2 \sum_{i=1}^{I} \mathrm{Re}\{\alpha_i g_i[t]\} . \quad (12.61)$$

The conjugate-subspace pursuit provides real models of real signals based on a complex dictionary, which enables the use of fast complex algorithms (e.g., the FFT) and simplifies phase estimation [12.19, 46]. Note however that this formulation only holds when $\boldsymbol{g}$ and $\boldsymbol{g}^*$ are linearly independent; other dictionary functions must be evaluated separately using appropriate selection metrics for comparison with the conjugate subspaces.

Since MP is a greedy algorithm, it does not necessarily arrive at the optimal global solution. Equations (12.50) and (12.54) indicate that the error of pursuit-based models decreases as the model order increases; indeed, the model converges to the signal in the mean-square sense [12.45]. However, for some model order I, there may be other I-term expansions based on the same dictionary that have a smaller modeling error.

Several variations of MP that attempt to find more-accurate solutions have been described [12.19, 48–52]. Such methods generally involve an orthogonalization at each step of the pursuit: in *orthogonal matching pursuit* [12.48], a new atom for the decomposition is selected at each iteration using the same metric as in MP, but after the selection the original signal is projected onto the current model subspace to minimize the modeling error; in *order-recursive matching pursuit* (ORMP) [12.49–51], also developed as *forward-orthogonal matching pursuit* [12.19] and *optimized orthogonal matching pursuit* [12.53], at each iteration the atom is selected so as to minimize the error for the resulting model subspace, which is equivalent to using the MP criterion and then orthogonalizing the dictionary to the selected atom [12.19]. These orthogonalization schemes typically outperform basic MP, but improved performance is not guaranteed; the resulting models may actually have a larger error than the basic MP model, and the additional computation required for these approaches may be a hindrance for real-time modeling.

One strategy for maintaining fast computation and improving the performance of the basic MP model is simply to accept the decomposition provided by MP and project the signal onto the model subspace after the pursuit is complete [12.54, 55]. This post-orthogonalization, or terminal back-projection, never yields a worse model than the basic MP, and entails significantly less computation than OMP or ORMP.

12.2.5 Sinusoidal Matching Pursuits

In many cases, e.g., some image coding schemes, the matching pursuit modeling algorithm is applied to the entire signal in question. Such an approach poses problems for natural speech and audio signals in that the pursuit search is computationally unwieldy for long-duration signals; it is clearly inapplicable for online applications such as speech coding for real-time communication. In order to handle general speech and audio, then, a sliding-window framework will be used for the sinusoidal modeling pursuit presented in this section. The idea in the analysis is to use MP to model windows of the signal as sums of windowed sinusoids; synthesis of the signal reconstruction is then carried out via overlap-add of these modeled windows. This approach can be specified mathematically as follows. Let $x_l[n]$ be the l-th windowed segment of the signal, namely $x_l[n] = w[n]x[n + lL]$, where $w[n]$ is a window of length N applied to the signal with a stride of L. OLA signal reconstruction from the windowed segments is given by

$$\hat{x}[t] = \sum_l x_l[t - lL] = x[t] \sum_l w[t - lL] , \quad (12.62)$$

which shows that $w[n]$ must satisfy the overlap-add property of (12.11) to avoid amplitude modulation artifacts in the reconstruction. For MP-based OLA modeling, each segment $x_l[n]$ is modeled using MP. Then, the convergence of the segment models implies a convergence of the OLA reconstruction to the original signal; and, if each segment is modeled exactly, perfect reconstruction is achieved.

The use of matching pursuit for sinusoidal modeling has been explored in various contexts in the literature. In [12.39], which predates the formalization of MP in [12.45], correlation is used as a metric for estimation of the amplitudes, frequencies, and phases of sinusoidal components for overlap-add synthesis. In [12.19,46,55], conjugate-subspace pursuit based on a dictionary of complex sinusoids is used to derive signal models. MP is used for fast parameter estimation in [12.56]; in [12.57], an approach related to MP-based OLA sinusoidal modeling is presented; sinusoidal speech coding based on MP is considered in [12.58]. Psychoacoustic metrics for sinusoidal matching pursuit of audio signals are described in [12.59, 60]. While such metrics have proven effective for low-rate parametric audio coding, we adhere to a mean-squared error metric in this treatment.

12.2.6 Sinusoidal Analysis

In this section, we expand on the MP-based OLA sinusoidal model described in [12.55], wherein the object of the pursuit is to find a set of windowed sinusoids to accurately model each windowed segment. The MP dictionary contains the functions

$$g[k,n] = \bar{w}[n]\,\mathrm{e}^{\mathrm{i}\omega_k n} \tag{12.63}$$

where k is a frequency index $(0 \le k \le K-1)$, $\omega_k = 2\pi k/K$, and $\bar{w}[n]$ is a normalized version of the N-point analysis window: $\bar{w}[n] = \frac{w[n]}{\|w[n]\|}$. The dictionary has K elements; completeness ($K \ge N$) is required for convergence of the pursuit, and overcompleteness ($K > N$) improves the accuracy of low-order models [12.19,45,46]. In the above formulation, a local time index n is used instead of the global index t used in Sect. 12.2.4 to emphasize that the signal is being modeled on a per-window basis instead of globally; note that this is analogous to the local time index used in practical sliding-window STFT implementations (as explained in Sect. 12.1.1). The implications of using a global versus a local time index in the modulation of sinusoidal atoms for MP are discussed in [12.46], especially with regard to increased computational requirements for the pursuit in the global case.

The MP dictionary defined in (12.63) consists of complex exponentials. Since the goal is to model a real-valued audio signal, the conjugate-subspace pursuit algorithm is applicable, and for this dictionary, the formulation can be greatly simplified. To begin with, consider the dictionary cross-correlation $\Gamma[k,m] = \langle g[k,n], g[m,n]\rangle$; this can be computed as follows:

$$\begin{aligned}\Gamma[k,m] &= \sum_{n=0}^{N-1} \bar{w}[n]^2\,\mathrm{e}^{-\mathrm{i}2\pi(k-m)n/K} \\ &= W_K[k-m]\,,\end{aligned} \tag{12.64}$$

where $W_K[k] = \mathcal{F}_K\left\{\bar{w}[n]^2\right\}$. The expression $\mathcal{F}_K\{\}$ denotes the K-point DFT; in (12.64), note that the summation terms from $n = N$ to $K-1$ are zero by definition of the window. The conjugate cross-correlation $\langle g[k,n], g[k,n]^*\rangle$ can then be expressed in terms of $W_K[k]$:

$$\begin{aligned}c[k] &= \langle g[k,n], g^*[k,n]\rangle \\ &= \Gamma[k,-k] \\ &= W_K[2k]\,.\end{aligned} \tag{12.65}$$

The correlation term $\langle g[k,n], r_i[n]\rangle$ can also be expressed in the DFT domain:

$$\begin{aligned}R_i[k] &= \langle g[k,n], r_i[n]\rangle \\ &= \sum_{n=0}^{N-1} \bar{w}[n] r_i[n]\,\mathrm{e}^{-\mathrm{i}2\pi nk/K} \\ &= \mathcal{F}_K\left\{\bar{w}[n] r_i[n]\right\}\,.\end{aligned} \tag{12.66}$$

Given the quantities $c[k]$ and $R_i[k]$, the optimal expansion coefficients given in (12.57) for the pair $\{g[k,n], g^*[k,n]\}$ can be specified for the sinusoidal modeling case as:

$$\begin{pmatrix}\alpha_i[k] \\ \alpha_i^*[k]\end{pmatrix} \quad \text{with} \quad \alpha_i[k] = \frac{(R_i[k] - c[k]R_i[k]^*)}{1-|c[k]|^2}\,. \tag{12.67}$$

Note that these coefficients are complex and include both the amplitude and phase of the conjugate-subspace components.

The conjugate-subspace pursuit optimization given in (12.58) and (12.59) can be specified for the sinusoidal pursuit as

$$g_i[k,n] = \arg\max_{g\in D} \Psi_i[k]\,, \tag{12.68}$$

where the optimization metric $\Psi_i[k]$ is only a function of $R_i[k]$ and $c[k]$:

$$\Psi_i[k] = 2\,\mathrm{Re}\{R_i[k]^*\alpha_i[k]\}\,. \tag{12.69}$$

Note that when $|c[k]| = |W_K[2k]| = 1$, the denominator in (12.67) is zero. Since $\bar{w}[n]$ has unit norm, $W_K[0] = W_K[K] = 1$ so that $|c[k]| = 1$ at $k = 0$ and $k = \frac{K}{2}$, namely when $g[k, n]$ is purely real. In this case, the conjugate-subspace formulation of the pursuit is not applicable since $g[k, n]$ and its conjugate are linearly dependent (identical, indeed). To account for this, each stage of the pursuit must compare $\Psi[k]$ for $1 \le k \le \frac{K}{2} - 1$ with the metrics $|\langle g[k, n], r_i[n]\rangle|^2$ for $k \in \{0, \frac{K}{2}\}$ to determine if a conjugate pair or a real function should be added to the model at that iteration; since $\Psi_i[k]$ and $|\langle g[k, n], r_i[n]\rangle|^2$ each quantify how much energy will be removed from the residual by the respective dictionary atoms, this is a fair comparison to determine the selection. Note that in practice components at $\frac{K}{2}$ are not considered and the k range of the pursuit is narrowed to focus the modeling to the frequency range of interest, e.g., an 8 kHz bandwidth for modeling speech signals. Note also that the k range for the pursuit covers only *positive* frequencies; unlike the basic pursuit metric, the conjugate-subspace pursuit metric is symmetric in k; with respect to search complexity, a sinusoidal dictionary of size K is essentially reduced in size by a factor of two by the conjugate-subspace formulation.

The conjugate-subspace pursuit metric given in (12.69) accounts for the spectral overlap of positive- and negative-frequency components. If magnitude correlations with complex sinusoids are used as the pursuit metric for modeling a real signal, parameter estimates derived from the positive-frequency spectrum can be biased by the negative-frequency components, especially at low frequencies; using such correlations is analogous to DFT peak-picking, where a similar bias occurs. The bias can be avoided by using conjugate-subspace pursuit, or quadrature models, which are functionally equivalent but less computationally efficient [12.39, 57]. An example of the estimation bias is depicted in Fig. 12.13, which shows the conjugate-pursuit metric of (12.69) and the correlation metric $|\langle \boldsymbol{g}, \boldsymbol{r}_i\rangle|^2$ for the first sinusoidal pursuit iteration for modeling a signal consisting of a single real sinusoid. In the simulation, the signal is of length $N = 128$, the dictionary is of size $K = 512$, and $w[n]$ is a Hanning window. The conjugate-pursuit metric yields a correct estimate of the sinusoidal parameters, while the correlation metric yields inaccurate estimates since its peak is at a lower frequency than the actual sinu-

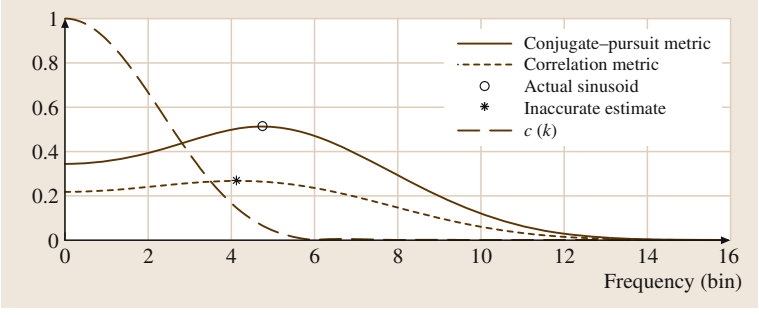


Fig. 12.13 Metrics for sinusoidal pursuit. For pursuit of a single real sinusoid (*open circle*), the conjugate-subspace pursuit metric exhibits a peak at the correct location and therefore results in a correct estimate of the sinusoid's parameters; the correlation metric of basic MP yields incorrect parameter estimates

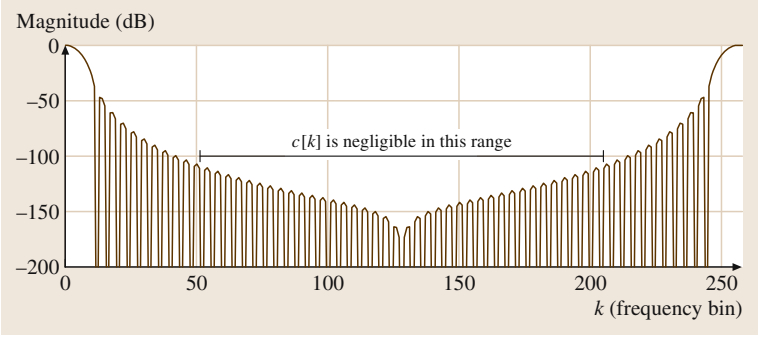


Fig. 12.14 Conjugate-pair cross-correlations for a Hanning-windowed sinusoidal dictionary with $N = 64$ and $K = 512$. At mid-range frequencies, a complex sinusoid and its conjugate can be reasonably approximated to be orthogonal, such that $c[k]$ can be neglected in (12.67)

soid due to the interaction of the negative-frequency component.

The extent of this interaction for the $N \times K$ pursuit dictionary is described by $c[k]$, the cross-correlation between a complex dictionary sinusoid of frequency $2\pi k/K$ and its conjugate. This conjugate cross-correlation $c[k]$ is also plotted in Fig. 12.13; at the frequency of the sinusoid in the signal, the conjugate components are not orthogonal and $c[k]$ is not negligible.

Despite the estimation bias incurred, some pursuit schemes described in the literature neglect the interaction of positive- and negative-frequency components; in these approaches, complex positive-frequency components are estimated using magnitude correlations between the real signal and a complex sinusoidal dictionary, and the model is subsequently made real by adding a corresponding conjugate at the end of each pursuit iteration [12.59, 60]. This approach is equivalent to assuming that $c[k] \approx 0$, which can be seen in Fig. 12.14 to be a reasonable approximation for mid-range frequencies. Since $W_K[K] = W_K[0] = \sum_n \tilde{w}[n]^2$, $c[k] = W_K[2k]$ takes on significant values in the vicinity of both $k = 0$ and $k = K/2$, so estimation bias is incurred for both low- and high-frequency components if $c[k]$ is altogether neglected, and modeling performance is degraded with respect to the conjugate-subspace approach. An example of mean-square modeling performance is given in Fig. 12.15, which shows the convergence of three pursuit-based sinusoidal models as a function of model order for a speech signal; *basic pursuit* refers to modeling a real signal with complex sinusoids and *basic real pursuit* refers to adding a conjugate component at the end of each iteration. Using the conjugate-subspace pursuit approach tends to improve the rate of model convergence, but this modeling improvement is sacrificed in the cited methods since adhering to the $c[k] \approx 0$ assumption (for all frequencies) facilitates the incorporation of psychoacoustic modeling criterion [12.59, 60].

In addition to the pursuit metric, it is necessary to formulate the correlation update for the sinusoidal pursuit. For the sinusoidal dictionary, the efficient update in (12.53) can be specified as follows. For $g[k_i, n]$ chosen at the i-th iteration, the new residual at stage $i+1$ is

$$r_{i+1}[n] = r_i[n] - \alpha_i g[k_i, n] - \alpha_i^* g^*[k_i, n] , \quad (12.70)$$

where the shorthand $\alpha_i = \alpha_i[k_i]$ is being used; note that the initial condition for modeling the l-th signal block is $r_1[n] = x_l[n] = w[n]x[n + lL]$. Taking the correlation of both sides of (12.70) with a dictionary atom $g[k, n]$, the correlation of the updated residual with the dictionary

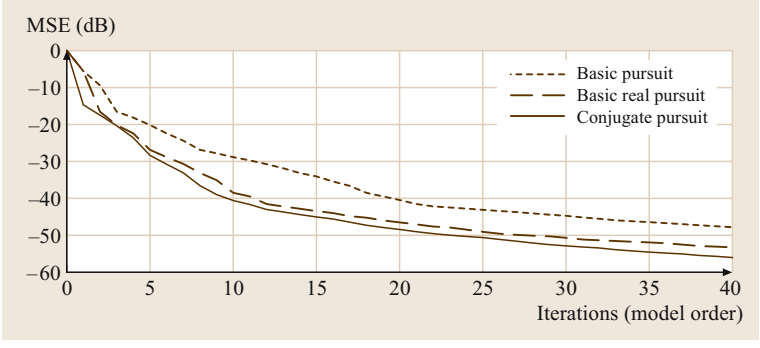


Fig. 12.15 Comparison of sinusoidal model convergence for basic pursuit, basic pursuit with a conjugate component added at the end of each iteration, and conjugate-subspace pursuit. In the simulation, $N = 256$ and $K = 1024$

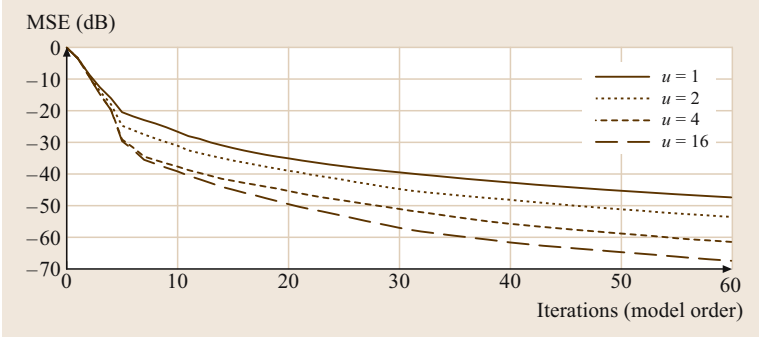


Fig. 12.16 Sinusoidal pursuit convergence as a function of model order for different frequency-domain oversampling factors. In the simulation, the block size is $N = 1024$ and the DFT size is $K = uN$, where u is indicated in the legend. The rate of model convergence increases as the dictionary size is increased

can be written as

$$R_{i+1}[k] = R_i[k] - \alpha_i W_K[k-k_i] - \alpha_i^* W_K[k+k_i] \,. \tag{12.71}$$

With this result, we see that all of the computation needed for the proposed sinusoidal pursuit can be carried out in the K-point DFT domain. The choice of K, namely the size of the dictionary, dictates the convergence rate; a larger dictionary, i. e., a larger oversampling ratio K/N for the DFT, leads to faster convergence of the signal model. Examples of sinusoidal model convergence for various oversampling ratios are depicted in Fig. 12.16. The modeling improvement achieved by using a larger K comes at the cost of additional model data; more bits are required for indexing into larger dictionaries.

The sinusoidal pursuit algorithm for modeling a signal frame $x_l[n] = w[n]x[n+lL]$ can be summarized as follows, starting with $i = 1$:

1. Compute $R_1[k] = \mathcal{F}_K\{\bar{w}[n]x_l[n]\}$.
2. Compute $\alpha_i[k] = \frac{(R_i[k] - c[k]R_i[k]^*)}{1-|c[k]|^2}$.
3. Compute $\Psi_i[k] = 2\,\mathrm{Re}\{R_i[k]^*\alpha_i[k]\}$.
4. Find $k_i = \arg\max_k \Psi_i[k]$.
5. Add $g[n, k_i]$ and $\alpha_i[k_i]$ to the model.
6. Unless a stopping criterion (such as model error) is met, compute $R_{i+1}[k] = R_i[k] - \alpha_i W_K[k-k_i] - \alpha_i^* W_K[k+k_i]$ and iterate from step 2.

The pursuit iterations are discontinued when the signal model meets some criterion such as an error threshold or a maximum allowed order; in coding applications, a psychoacoustic constraint or a bit budget might be incorporated. Then, the model coefficients can be refined by projecting the signal onto the model subspace in a terminal back-projection step. This post-orthogonalization can be carried out by an explicit projection, which would take the form of (12.55) with G containing the model functions, or by a gradient descent, which is more cost-efficient than an explicit projection.

Note that, if the sinusoidal model does not fit the signal well, a higher model order is required to reach a given modeling error threshold; an obvious example of such a mismatched signal is white noise, since it has no coherent pseudo-sinusoidal structure. To account for such cases, it is useful to include additional model components as in sinusoid-plus-residual models [12.19,38,61–64] and sinusoids-plus-transients-plus-noise representations [12.41]. Such hybrid models have been shown to effectively overcome the problem of model mismatch. Multiresolution analysis–synthesis frameworks have also proven useful for improving the effectiveness of the sinusoidal model for representing arbitrary input signals [12.55, 65].

12.2.7 Sinusoidal Synthesis

Synthesis in the MP-based sinusoidal model is achieved by overlap-add of the modeled segments: $\hat{x}[t] = \sum_l \hat{x}_l[t-lL]$. For each segment, the signal model derived by the conjugate-subspace sinusoidal pursuit has the following form, assuming only conjugate pairs are selected for the model (no components at $k = 0$ or $k = \frac{K}{2}$ are selected):

$$\begin{aligned}\hat{x}_l[n] &= \sum_{i=1}^{I_l} \alpha_{l,i}\bar{w}[n]\mathrm{e}^{\mathrm{i}2\pi n k_{l,i}/K} \\ &\quad + \sum_{i=1}^{I_l} \alpha_{l,i}^*\bar{w}[n]\mathrm{e}^{-\mathrm{i}2\pi n k_{l,i}/K} \\ &= 2\bar{w}[n]\sum_{i=1}^{I_l} |\alpha_{l,i}| \cos\left(\frac{2\pi n k_{l,i}}{K} + < \alpha_{l,i}\right) \,.\end{aligned} \tag{12.72}$$

Efficient synthesis for this model can be achieved using the inverse FFT. Letting $\bar{W}[k] = \mathcal{F}_K\left\{\bar{w}[n]\right\}$, the K-point DFT of $\hat{x}_l[n]$ is

$$\begin{aligned}\hat{X}_{l,K}[k] &= \sum_{i=1}^{I_l} \alpha_{l,i}\bar{W}_K[k-k_{l,i}] \\ &\quad + \alpha_{l,i}^*\bar{W}_K[k+k_{l,i}] \,.\end{aligned} \tag{12.73}$$

For a K-point synthesis DFT, synthesis is achieved by accumulating weighted and shifted versions of the normalized window spectrum $\bar{W}[k]$. If integer oversampling $\mu = \frac{K}{N}$ is used in the analysis, i. e., if the analysis DFT is zero-padded with $(\mu-1)N$ zeros, the oversampling redundancy can be removed to reduce the synthesis complexity by directly deriving the N-point DFT of $\hat{x}_l[n]$:

$$\begin{aligned}\hat{X}_{l,N}[l] &= \sum_{i=1}^{I_l} \alpha_{l,i}\bar{W}_K[\mu l - k_{l,i}] \\ &\quad + \alpha_{l,i}^*\bar{W}_K[\mu l + k_{l,i}] \,.\end{aligned} \tag{12.74}$$

An N-point inverse DFT is then used for synthesis.

The MP-based overlap-add sinusoidal model uses the same window $w[n]$ for analysis and synthesis. It is chosen to have a sparse spectral representation, i. e., $\bar{W}_K[k]$ and $W_K[k]$ both consist of a concentrated main lobe and low-level sidelobes. For $K = N$, general Blackman–Harris or cosine-sum windows are good

choices since their spectra have only a few nonzero values, which simplifies the computation of the synthesis spectra. When $K > N$, the window spectra have many low-level nonzero values in the sidelobe regions. To reduce the synthesis complexity, these values can be excluded in the frequency-domain synthesis of (12.74) at the cost of minor errors [12.18]. In the analysis, the sidelobe regions of $W_K[k]$ and $c[k]$ can be excluded to reduce the computational cost, but this counteracts some of the advantages of analysis by synthesis and may lead to degraded parameter estimation and the identification of sidelobes as sinusoidal signal components.

Since the analysis and synthesis for the sinusoidal matching pursuit algorithm can be carried out in the DFT domain, this modeling approach fits into the framework depicted in Fig. 12.12 as anticipated earlier. In the parametric model block of the figure, each time-localized spectrum of the STFT $X[k, l]$ is approximated via the sinusoidal pursuit; the synthesis accumulates the spectral representations of windowed sinusoids to construct $\hat{X}[k, l]$, which is provided to the synthesis filter bank to generate the time-domain output. From this perspective, the synthesis window $g[n]$ is a rectangular K-point window, or, for the integer oversampling case described above, $g[n]$ is an N-point rectangular window. Alternatively, the synthesis window $g[n]$ can be selected to be the modeling window $\bar{w}[n]$, in which case the parametric model block simply provides a sparse set of inputs to the synthesis filter bank. Only the synthesis filters corresponding to the selected sinusoidal model components are fed nonzero inputs at time l, and these inputs are given by the corresponding component weights deter-

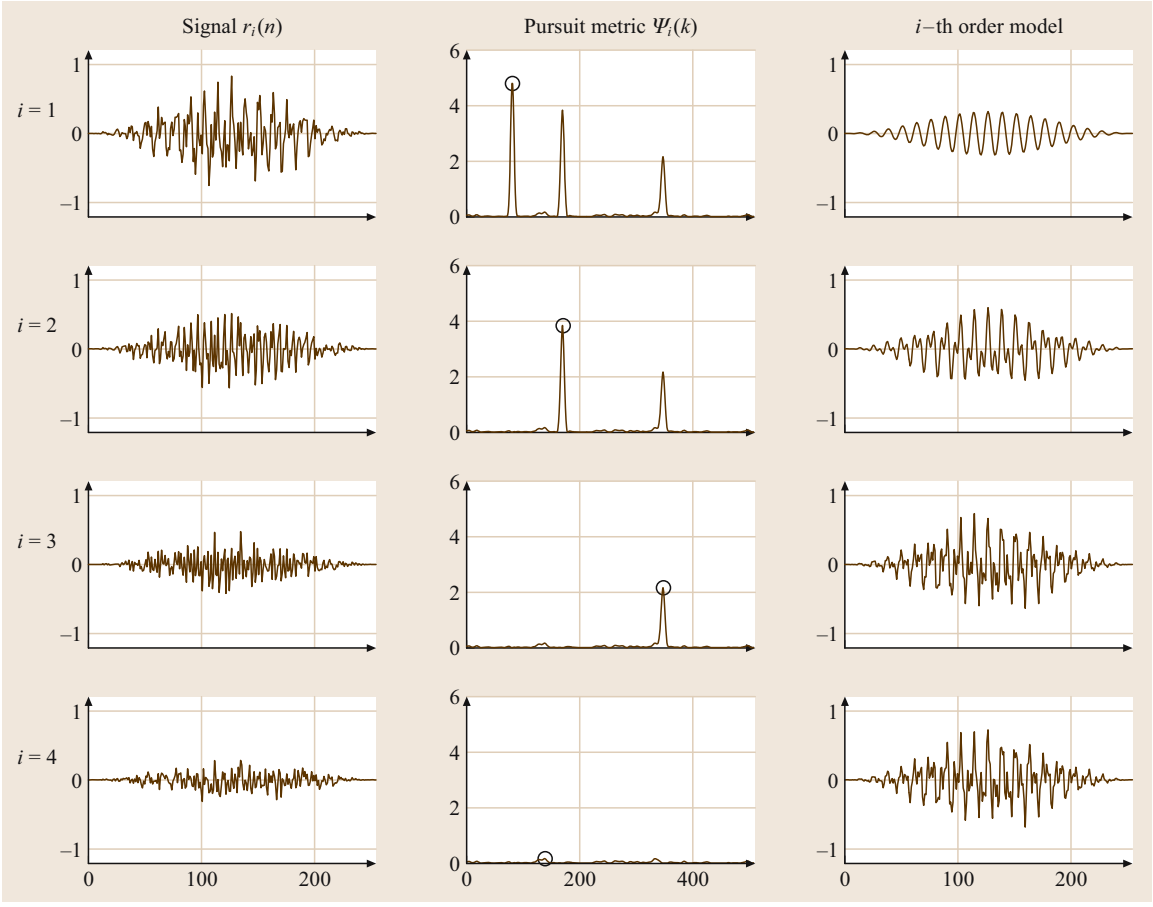


Fig. 12.17 Iterations of sinusoidal matching pursuit for a synthetic multi-sinusoid signal. The first column shows the residual $r_i[n]$ at the start of the iteration; the second column shows the MP metric $\Psi_i[k]$ from (12.69) and its maximum (*open circles*); the third column shows the i-term signal model after the i-th iteration

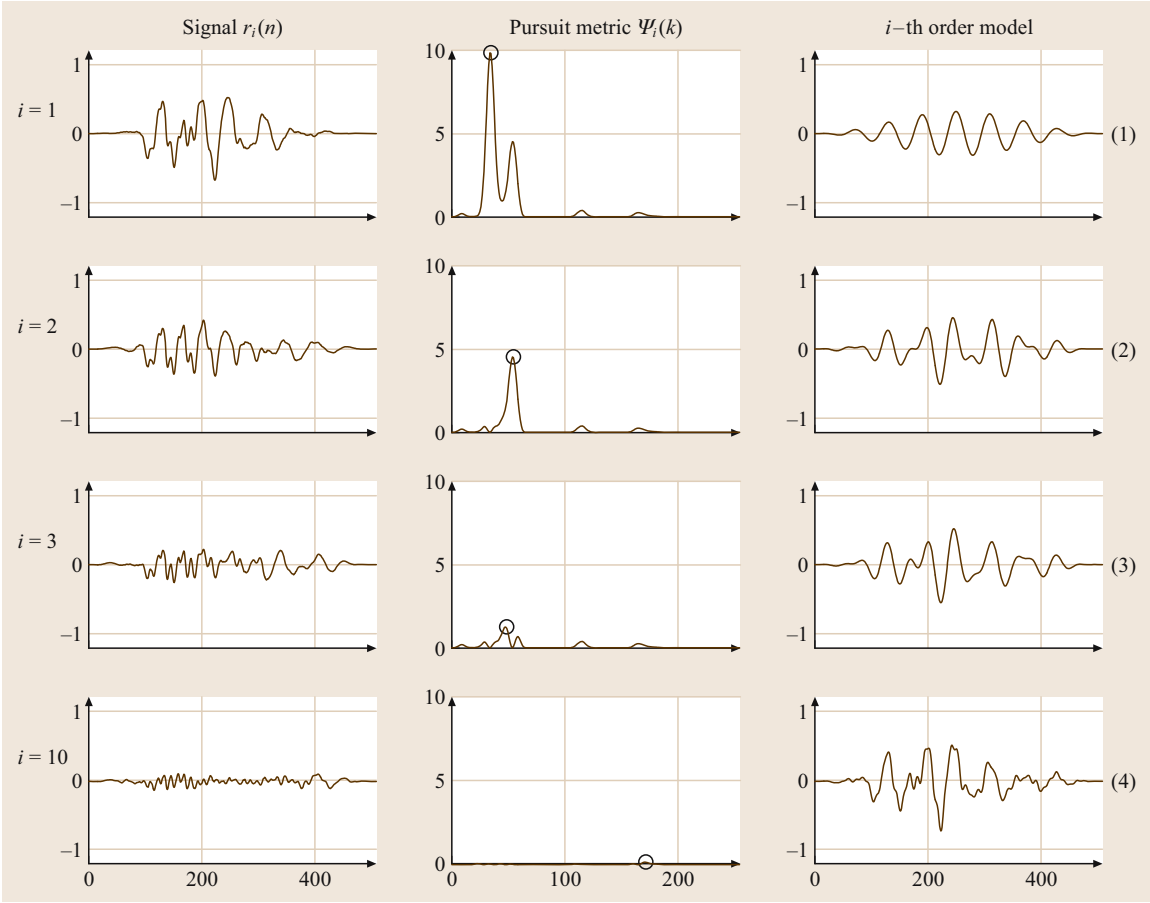


Fig. 12.18 Iterations of sinusoidal matching pursuit for a speech signal. The first column shows the residual $r_i[n]$ at the start of the iteration; the second column shows the MP metric $\Psi_i[k]$ from (12.69) and its maximum (*open circles*); the third column shows the i-term signal model after the iteration. Rows 1–3 show the first three pursuit iterations; the fourth row shows the tenth iteration

mined by the pursuit. In this interpretation, the synthesis filter bank accumulates the weighted windowed sinusoids. In practice, the spectral synthesis approach is used since it leads to more-efficient implementations; the synthesis filter bank is not explicitly realized in the time domain but rather as an inverse DFT and overlap-add procedure.

Figure 12.17 depicts four iterations of the sinusoidal matching pursuit algorithm for a synthetic multi-sinusoid signal. The first column shows $r_i[n]$, the time-domain signal that the pursuit is attempting to match at iteration i. Recall from the algorithm description that this time-domain residual signal is not computed explicitly since the required metrics can all be maintained in the frequency domain; it is simply shown here to illustrate the modeling behavior. The second column in the figure shows the conjugate-subspace pursuit metric $\Psi_i[k]$; the circle indicates the peak identified by the metric maximization search. Finally, the third column depicts the signal model reconstruction at the end of the i-th iteration. The modeling of a speech signal is depicted in Fig. 12.18; note that the fourth row depicts the tenth iteration, and that the primary structure of the original signal is clearly captured by these early stages of the pursuit. As will be discussed in Sect. 12.3, compactly representing the salient features of the signal in such a way enables efficient coding and modification.

12.3 Speech Modification

Modification of speech signals is a wide-ranging subject covering applications from improving intelligibility for the sake of communication to changing ones persona for entertainment or virtual reality. At the signal level, speech modification involves such processes as noise removal, dereverberation, equalization, time scaling and pitch shifting. In this section, we discuss the use of the STFT and sinusoidal representations to carry out such signal-level modifications and provide relevant pointers to the literature for additional algorithmic details.

12.3.1 Comparing the STFT and Sinusoidal Models

Many signal modification schemes based on the STFT and/or sinusoidal modeling have been described in the literature. In early approaches based directly on the STFT subband signals $X[k,l]$, modifications were restricted to a relatively rigid framework because the signal is being modeled in terms of subbands that interact in complicated ways in the reconstruction process; this restrictive framework is namely that a modification is proposed, e.g., a time-varying spectral multiplication, and then the effect of the modification on the output signal is formulated [12.1, 11, 12, 14]. This approach is effective for understanding the operating principles of the STFT, but is somewhat removed from the desired end of carrying out an intended modification on the original signal.

In later approaches, modifications based only on the STFT magnitude were considered; the magnitude is first modified and then a phase that will minimize synthesis discontinuities is derived [12.20]. This removal of the phase essentially results in a pseudo-parametric representation that is more flexible than the complex subband signals. For some modifications, the phase of the original signal can be reused with the modified magnitude without introducing objectionable artifacts [12.66]; for others, such as time-scale modification, a consistent phase must be generated using the original phase and various continuity and validity constraints [12.19, 36].

The usefulness of the sinusoidal model is underscored by the difficulties of some STFT-based modifications, namely those of formalizing arbitrary modifications in a rigid synthesis framework and of generating an appropriate phase for magnitude-only modifications. Since it is fully parametric, the synthesis of a modified sinusoidal model is not constrained in the same way as an STFT; the model parameters are more malleable than STFT subband data in that there are no consistency or validity concerns or frame-boundary discontinuity issues.

In the following, we describe several classes of modification and highlight the differences between the STFT and sinusoidal approaches.

12.3.2 Linear Filtering

Linear filtering is a ubiquitous signal processing operation, and it is well known that the time-domain convolution operation of linear filtering corresponds to multiplication in the frequency domain. In the DFT domain, this principle can be stated as follows:

$$x[n] * f[n] \overset{\text{DFT}}{\Longleftrightarrow} X_K[k]F_K[k]\,, \tag{12.75}$$

where $X_K[k]$ and $F_K[k]$ are the K-point DFTs of the input sequence $x[n]$ and the filter impulse response $f[n]$, respectively; the DFT of the convolution is the product of the DFTs of the input signal and the filter impulse response. This property only holds exactly, however, if the DFT length satisfies

$$K \geq N_x + N_f - 1\,, \tag{12.76}$$

where N_x and N_f are the signal length and the impulse response length, respectively. If K is not large enough, the spectral multiplication introduces time-domain aliasing.

DFT-based filtering is of interest since the use of the FFT leads to implementations that require less computation than direct time-domain convolution. The signal and filter are transformed to the frequency domain via the FFT, multiplied, and then inverse-transformed back to the time domain. Given the DFT size requirement, however, this approach is not practical for arbitrarily long signals; to avoid time-domain aliasing, a correspondingly large FFT would be required. To enable the practical DFT-based filtering of arbitrarily long signals, the input signal is windowed into short segments, and the filtering operation is applied individually to each segment. In the *overlap-add* approach, the window length and DFT size are selected such that (12.76) holds for each segment. Then, each segment is individually filtered via spectral multiplication; each filtered spectrum is subsequently inverse-transformed and the time-domain outputs are combined via overlap-add with a rectangular synthesis window of length K. An alternative approach is the *overlap-save* method, where

the size constraint is less restrictive than (12.76); some time-domain aliasing is incurred, but corrupted output samples are discarded. More details on filter implementation using FFTs are available in the literature [12.67].

The overlap-add filtering approach fits readily into the modulated filter bank interpretation of the STFT shown in Fig. 12.2; a linear filter can be realized by multiplication of the subband signals $X[k, l]$ with the filter spectrum $F[k]$, and synthesis is carried out with a rectangular $v[n]$ of length K. Similarly, such overlap-add filtering can be employed in the parametric modeling scheme of Fig. 12.12. The approximate spectrum $\hat{X}[k, l]$ constructed from the sinusoidal parameters can be multiplied by $F[k]$ to achieve *exact* filtering of the output signal; this is not exact filtering of the input signal since the parametrically modeled spectrum is an approximation. An alternate approach is to carry out the filtering in the parametric domain by multiplying the sinusoidal model component amplitudes by the value of the filter spectrum at the corresponding frequencies; sinusoidal synthesis is then carried out based on these scaled model coefficients. This method yields *approximate* filtering of the output signal with significantly less computation than the former exact filtering [12.68].

Applications of linear filtering in speech processing include equalization of communication channels, noise suppression, and spatialization in binaural rendering, among many others such as speech enhancement, which will be discussed briefly in the following section.

12.3.3 Enhancement

In signal processing, the term *enhancement* typically refers to improving the signal-to-noise ratio or the intelligibility of a speech signal. A review of this large field is beyond the scope of this chapter, but it merits mentioning that many enhancement approaches involve adaptive linear filtering. Others are based on suppression of spectral components, either using noise estimates [12.69] or operating on the assumption that small spectral values correspond primarily to unwanted noise. This latter notion of signal denoising by thresholding the coefficients of the signal representation has been explored in detail in the literature [12.70, 71]; the idea is that large model coefficients capture the salient signal features while small model coefficients are essentially more representative of noise and should be discarded. Since a sinusoidal model captures the key features of typical speech signals, this notion suggests that sinusoidal modeling would prove effective for enhancement of degraded speech since the noise would not be captured in a compact sinusoidal representation. In an enhancement experiment, however, the sinusoidal model proved somewhat ineffective in this regard, so further investigation of this concept is needed [12.72].

12.3.4 Time-Scale Modification

In some scenarios, it is useful to change the duration of a speech or audio signal without altering its general character; interpolating the time-domain signal samples is insufficient since a spectral shift (e.g., a pitch shift) is incurred (if the same sampling rate is used). Techniques for high-quality time-scale modification are thus of interest; applications range from audio–video synchronization to slowing down a signal for music instruction or transcription or for improving the identifiability of the signal content. A wide range of algorithms have been proposed in the literature, including STFT magnitude modification followed by phase estimation [12.34, 35], sinusoidal modeling methods [12.41, 43, 65, 73, 74], and analyzing the signal for time-domain regions, e.g., pitch periods, which can be spliced out of the signal for time-scale compression or repeated for time-scale expansion [12.19, 75]; many other methods have also been considered [12.31], but a full survey is beyond the scope of this chapter.

In [12.19], the application of compact signal models to signal modification is discussed at length. Fundamentally, the idea is that signal modifications can be realized by modifying the components of a model of the signal. The sinusoidal model is particularly amenable to this approach because modifications of general interest are easy to carry out on sinusoids. For the case of time scaling, it is simple to increase or decrease the duration of a sinusoid – so if a signal is modeled as a sum of sinusoids, it becomes relatively simple to carry out time scaling on the entire signal. One caveat to note is that in some time-scaling scenarios it is important to preserve the rate of variation in the amplitude envelope of the signal, i. e., the signal dynamics, but this can be taken into account [12.74]. One sinusoidal approach described in the literature involves segmenting the signal into transient and sinusoidal regions; high-quality time scaling is achieved by translating the transient segments and then appropriately interpolating the sinusoidal parameters to shorten or lengthen the regions between transients [12.41, 65].

Sinusoidal modeling has been applied successfully to speech coding and postprocessing for packet-switched networks [12.76]. In the event of a lost packet, it is necessary for the decoder to construct the time-domain

audio output without the coded data contained in the missing packet, i. e., based only on the information in the adjacent received packets. In a sinusoidal modeling scheme, partials of the speech signal can be matched and interpolated across these adjacent frames to create a consistent time-domain replacement for the missing data. This can be thought of as time scaling the adjacent sinusoidal data to fill in the time-domain gap resulting from the missing packet.

12.3.5 Pitch Modification

Changing the pitch of a speech signal is another modification of interest. This is often thought of as the dual of time-scale modification; the goal is to shift the pitch of the signal without changing the duration, whereas in time-scaling the object is to change the duration without altering the pitch.

Signals with a well-defined pitch have a periodic or pseudoperiodic structure in the time domain; in the frequency domain, such signals have spectral peaks at or near integer multiples of a fundamental frequency. One approach to pitch modification, then, is to move the spectral peaks such that they are related by a new fundamental frequency and thus result in a new perceived pitch in the resynthesized signal. STFT-domain methods based on such peak moving have been discussed in the literature, and methods for establishing an appropriate phase for the shifted peaks have been considered [12.35, 36].

Given its parameters, a sinusoidal signal model of course enables modification of the signal's pitch. Most simply, pitch modification can be carried out by scaling the frequencies prior to synthesis. For voice applications, however, this results in unnatural reconstructed speech. Natural pitch transposition can be achieved by interpreting the sinusoidal parametrization as a source-filter model and carrying out formant-corrected pitch shifting as described below.

The sinusoidal model parametrization of a signal inherently includes a description of the signal's spectral envelope, which can be interpreted as a time-varying filter in a source-filter model in which the source is a sum of unweighted sinusoids. In voice applications, the filter corresponds to the vocal tract and the source represents the glottal excitation. Using this analogy, modifications can incorporate the physical underpinning that a pitch shift in speech is produced primarily by a change in the rate of glottal vibration and not by a change in the vocal tract shape or its resonances. To achieve natural pitch shifting of speech or the singing voice using the sinusoidal model, then, the spectral envelope must be preserved in the modification stage so as to preserve the formant structure of the vocal tract. Pitch modification is thus carried out by scaling the frequency parameters of the excitation sinusoids and then deriving new amplitudes for these pitch-scaled sinusoids by sampling the spectral envelope at the new frequencies, using interpolation of the envelope for frequencies that do not fall on the spectral bins. Such formant correction can be used in STFT-based peak-moving schemes to similarly improve the naturalness of the modification.

12.3.6 Cross-Synthesis and Other Modifications

A considerable array of signal manipulations beyond time-scale and pitch modification are enabled by STFT and sinusoidal representations [12.19,35]. Many of these are beyond the scope of this section, but some deserve brief mention for speech processing applications. For instance, the source-filter interpretation of the sinusoidal model is useful for a variety of spectral manipulations. In general, any sort of time-varying filtering can be carried out by modifying the amplitudes in the sinusoidal model representation. For instance, the formants in the spectral envelope can be adjusted to yield gender modifications; by moving the formants down in frequency, a female voice can be transformed into a male voice, and vice versa. Also, the amplitude ratios of odd and even harmonics in a pitched signal can be adjusted to achieve interesting signal changes.

So far the modifications we have discussed are operations carried out a single original signal; *cross-synthesis* refers to methods in which a new signal is created via the interactions of two or more original signals. An example of cross-synthesis is a source-filter model combination created by using the source from one signal's model and the filter from another signal's model, for instance exciting the vocal-tract filter estimated from a male voice by the glottal source estimated from a female voice. Such cross-synthesis is useful for voice persona modification.

The sinusoidal model enables compelling cross-synthesis modifications since the parameters directly describe perceptually important signal qualities such as the pitch as well as the shape and evolution of the spectral envelope. One immediate example of a modification is interpolation between the sinusoidal parameters of two sounds; this yields a hybrid signal perceived as a coherent merger of the two original sounds, and not simply a cross-fade or averaging. This type of modification

has received considerable attention for image *morphing*, which is carried out by parameterizing the salient features of an original image and a target image (such as edges or prominent regions) and creating a map between these parameterized features that can be traversed to construct a morphed intermediate image [12.77]. A related idea has been applied in the STFT domain for morphing between different audio signals [12.78].

12.3.7 Audio Coding with Decode-Side Modification

As described in Sect. 12.1.8, the STFT is ill-suited for audio coding since the subband data must be oversampled to achieve perfect reconstruction with an overlapping synthesis window. There is a data increase in the STFT-domain representation, which counteracts the key goal of signal compression. However, in the event that modification of the coded audio signal is required or desired at the decoder, it may be worthwhile to sacrifice the data rate for the added post-processing flexibility that an STFT representation provides over standard transform-coding representations such as the modified discrete cosine transform (MDCT). In [12.68], it is shown that using a signal representation amenable to modification can provide a significant reduction in the computational complexity of the decoder, especially for multichannel signals.

As for the sinusoidal model, using an efficient and flexible parametric representation for signal coding provides both a substantial data-rate improvement over the STFT as well as a further reduction in the decoder complexity since modifications can be carried out efficiently on the compact signal parameters. This benefit is one of the motivating factors for using sinusoidal models for low-rate coding of speech and audio signals. At very low data rates, where perceptually transparent coding of audio signals is not currently achievable, parametric coders using sinusoidal models can outperform subband/transform coders and have become part of modern audio compression standards [12.56, 79]. At higher rates, subband/transform coders are currently more effective than state-of-the-art parametric methods for near-transparent coding, although the malleability of a sinusoidal model provides an incentive for accepting a data-rate penalty [12.68, 80]; as signal modeling algorithms improve and parametric coding methods approach the maturity of transform coders, it can be expected that this data-rate penalty will become minimal or even negligible [12.41, 55, 57, 60, 64, 81]. As for pure speech coding, the situation is similar: low-rate sinusoidal coders have been proven effective, but not quite as effective as linear predictive coders [12.58, 82, 83]. However, the increased flexibility of the sinusoidal representation for desirable post-processing of the output signal is a strong incentive for adopting a sinusoidal coder in some scenarios.

References

12.1 L.R. Rabiner, R.W. Schafer: *Digital Processing of Speech Signals* (Prentice-Hall, Englewood Cliffs 1978)

12.2 D. Gabor: Theory of communication, J. IEE **93**(III-26), 429–457 (1946)

12.3 D. Gabor: Acoustical quanta and the theory of hearing, Nature **159**(4044), 591–594 (1947)

12.4 M. Vetterli, J. Kovačević: *Wavelets and Subband Coding* (Prentice-Hall, Englewood Cliffs 1995)

12.5 P.P. Vaidyanathan: *Multirate Systems and Filter Banks* (Prentice-Hall, Englewood Cliffs 1993)

12.6 T.F. Quatieri: *Discrete-Time Speech Signal Processing* (Prentice-Hall, Upper Saddle River 2002)

12.7 J. Cooley, J. Tukey: An algorithm for the machine calculation of complex Fourier series, Math. Comput. **19**(90), 297–301 (1965)

12.8 P. Duhamel, M. Vetterli: Fast Fourier transforms: a tutorial review and a state of the art, Signal Process. **4**(19), 259–299 (1990)

12.9 R. Schafer, L. Rabiner: Design and simulation of a speech analysis-synthesis system based on short-time Fourier analysis, IEEE Trans. Audio Electroacoust. **AU-21**(3), 165–174 (1973)

12.10 M.R. Portnoff: Implementation of the digital phase vocoder using the fast Fourier transform, IEEE Trans. Acoust. Speech **24**(3), 243–248 (1976)

12.11 J. Allen: Short term spectral analysis, synthesis, and modification by discrete Fourier transform, IEEE Trans. Acoust. Speech **25**(3), 235–238 (1977)

12.12 J. Allen, L. Rabiner: A unified approach to short-time Fourier analysis and synthesis, Proc. IEEE **65**(11), 1558–1564 (1977)

12.13 J.O. Smith: *Mathematics of the Discrete Fourier Transform (DFT)*, 2nd edn. (Booksurge, Seattle 2007), http://ccrma.stanford.edu/jos/mdft/

12.14 M.R. Portnoff: Time-frequency representation of digital signals and systems based on short-time

Fourier analysis, IEEE Trans. Acoust. Speech **28**(1), 55–69 (1980)
12.15 R. Crochiere: A weighted overlap-add method of short-time Fourier analysis/synthesis, IEEE Trans. Acoust. Speech **28**(1), 99–102 (1980)
12.16 F.J. Harris: On the use of windows for harmonic analysis with the discrete Fourier transform, Proc. IEEE **66**(1), 51–83 (1978)
12.17 A.H. Nuttall: Some windows with very good sidelobe behavior, IEEE Trans. Acoust. Speech **29**(1), 84–91 (1981)
12.18 X. Rodet, P. Depalle: Spectral envelopes and inverse FFT synthesis, Proc. 93rd Conv. of the Audio Eng. Soc. (1992), Preprint 3393
12.19 M. Goodwin: *Adaptive Signal Models: Theory, Algorithms, and Audio Applications* (Kluwer Academic, Boston 1998)
12.20 D. Griffin, J. Lim: Signal estimation from modified short-time Fourier transform, IEEE Trans. Acoust. Speech **32**(2), 236–243 (1984)
12.21 H.S. Malvar: *Signal Processing with Lapped Transforms* (Artech House, Boston 1992)
12.22 Z. Czetkovic: *Overcomplete Expansions for Digital Signal Processing* (Univ. California, Berkeley 1995), PhD Dissertation
12.23 Z. Czetkovic, M. Vetterli: Oversampled filter banks, IEEE Trans. Signal Process. **46**(5), 1245–1255 (1998)
12.24 H. Bölcskei, F. Hlawatsch: Oversampled filter banks: Optimal noise shaping, design freedom, and noise analysis, IEEE ICASSP, Vol. 3 (1997) pp. 2453–2456
12.25 F. Léonard: Referencing the phase to the centre of the spectral window. Why?, Mech. Syst. Signal Process. **2**(1), 75–90 (1997)
12.26 M. Bosi, R. Goldberg: *Introduction to Digital Audio Coding and Standards* (Kluwer Academic, Boston 2003)
12.27 J.P. Princen, A.B. Bradley: Analysis/synthesis filter bank design based on time domain aliasing cancellation, IEEE Trans. Acoust. Speech **34**(5), 1153–1161 (1986)
12.28 H.S. Malvar, D.H. Staelin: The LOT: Transform coding without blocking effects, IEEE Trans. Acoust. Speech **37**(4), 553–559 (1989)
12.29 H. Dudley: The vocoder, Bell Lab. Rec. **18**, 122–126 (1939)
12.30 J.L. Flanagan, R.M. Golden: Phase vocoder, Bell Syst. Tech. J. **45**(9), 1493–1509 (1966)
12.31 E. Moulines, J. Laroche: Non-parametric techniques for pitch-scale and time-scale modification of speech, Speech Commun. **16**(2), 175–205 (1995)
12.32 J.A. Moorer: The use of the phase vocoder in computer music applications, J. Audio Eng. Soc. **26**(1/2), 42–45 (1978)
12.33 M. Dolson: The phase vocoder: A tutorial, Comput. Music J. **10**(4), 14–27 (1986)
12.34 J. Laroche, M. Dolson: Improved phase vocoder time-scale modification of audio, IEEE Trans. Speech Audio Process. **7**(3), 323–332 (1999)
12.35 J. Laroche, M. Dolson: New phase-vocoder techniques for real-time pitch shifting, chorusing, harmonizing, and other exotic audio modifications, J. Audio Eng. Soc. **47**(11), 928–936 (1999)
12.36 J. Laroche, M. Dolson: About this phasiness business, Proc. IEEE Workshop on Applications of Signal Process. to Audio and Acoust. (1997)
12.37 R.J. McAulay, T.F. Quatieri: Speech analysis/synthesis based on a sinusoidal representation, IEEE Trans. Acoust. Speech **34**(4), 744–754 (1986)
12.38 X. Serra, J. Smith: Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition, Comput. Music J. **14**(4), 12–24 (1990)
12.39 E.B. George, M.J.T. Smith: Analysis-by-synthesis/overlap-add sinusoidal modeling applied to the analysis and synthesis of musical tones, J. Audio Eng. Soc. **40**(6), 497–516 (1992)
12.40 P. Depalle, G. Garcia, X. Rodet: Tracking of partials for additive sound synthesis using hidden Markov models, IEEE ICASSP, Vol. 1 (1993) pp. 225–228
12.41 S. Levine, J.O. Smith: A sines+transients+noise audio representation for data compression and time/pitch scale modifications, 105th Audio Eng. Soc. Conv. (1998), Preprint 4781.
12.42 M. Lagrange, S. Marchand, J.-B. Rault: Using linear prediction to enhance the tracking of partials, IEEE ICASSP, Vol. 4 (2004) pp. 241–244
12.43 E.B. George, M.J.T. Smith: Speech analysis/synthesis and modification using an analysis-by-synthesis/overlap-add sinusoidal model, IEEE Trans. Speech Audio Process. **5**(5), 389–406 (1997)
12.44 R.J. McAulay, T.F. Quatieri: Computationally efficient sine-wave synthesis and its application to sinusoidal transform coding, IEEE ICASSP, Vol. 1 (1988) pp. 370–373
12.45 S. Mallat, Z. Zhang: Matching pursuits with time-frequency dictionaries, IEEE Trans. Signal Process. **41**(12), 3397–3415 (1993)
12.46 M. Goodwin, M. Vetterli: Matching pursuit and atomic signal models based on recursive filter banks, IEEE Trans. Signal Process. **47**(7), 1890–1902 (1999)
12.47 G. Davis: *Adaptive Nonlinear Approximations* (New York University, New York 1994), PhD Dissertation
12.48 Y. Pati, R. Rezaiifar, P. Krishnaprasad: Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition, Conf. Record of the Twenty-Seventh Asilomar Conf. on Signals, Systems, and Comput., Vol. 1 (1993) pp. 40–44
12.49 S. Chen, J. Wigger: Fast orthogonal least squares algorithm for efficient subset model selection, IEEE Trans. Signal Process. **43**(7), 1713–1715 (1995)

12.50 B. Natarajan: Sparse approximate solutions to linear systems, SIAM J. Comput. **24**(2), 227–234 (1995)
12.51 S. Singhal, B. Atal: Amplitude optimization and pitch prediction in multipulse coders, IEEE Trans. Acoust. Speech **37**(3), 317–327 (1989)
12.52 J. Adler, B. Rao, K. Kreutz-Delgado: Comparison of basis selection methods, Conf. Record of the Thirtieth Asilomar Conf. on Signals, Systems, and Comput., Vol. 1 (1996) pp. 252–257
12.53 L. Rebollo-Neira, D. Lowe: Optimized orthogonal matching pursuit approach, IEEE Signal Proc. Let. **9**(4), 137–140 (2002)
12.54 G. Davis, S. Mallat, Z. Zhang: Adaptive time-frequency decompositions with matching pursuit, Opt. Eng. **33**(7), 2183–2191 (1994)
12.55 M. Goodwin: Multiscale overlap-add sinusoidal modeling using matching pursuit and refinements, IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (2001) pp. 207–210
12.56 H. Purnhagen, N. Meine, B. Edler: Speeding up HILN – MPEG-4 parametric audio coding with reduced complexity, 109th Audio Eng. Soc. Conv. (2000), Preprint 5177
12.57 K. Vos, R. Vafin, R. Heusdens, W. Kleijn: High-quality consistent analysis-synthesis in sinusoidal coding, 17th Audio Eng. Soc. Int. Conf. (1999) pp. 244–250
12.58 C. Etemoglu, V. Cuperman: Matching pursuits sinusoidal speech coding, IEEE Trans. Speech Audio Process. **11**(5), 413–424 (2003)
12.59 T.S. Verma, T. Meng: Sinusoidal modeling using frame-based perceptually weighted matching pursuits, IEEE ICASSP (1998)
12.60 R. Heusdens, R. Vafin, W.B. Kleijn: Sinusoidal modeling using psychoacoustic-adaptive matching pursuits, IEEE Signal Proc. Lett. **9**(8), 262–265 (2002)
12.61 J. Laroche, Y. Stylianou, E. Moulines: HNM: A simple, efficient harmonic + noise model for speech, IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (1993) pp. 169–172
12.62 M. Goodwin: Residual modeling in music analysis-synthesis, IEEE ICASSP, Vol. 2 (1996) pp. 1005–1008
12.63 K. Hamdy, M. Ali, A. Tewfik: Low bit rate high quality audio coding with combined harmonic and wavelet representations, IEEE ICASSP, Vol. 2 (1996) pp. 1045–1048
12.64 A. Oomen, A. Den Brinker: Sinusoids plus noise modeling for audio signals, 17th Audio Eng. Soc. Int. Conf. (1999) pp. 226–232
12.65 S. Levine, T. Verma, J. Smith: Alias-free, multiresolution sinusoidal modeling for polyphonic wideband audio, IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (1997)
12.66 M. Goodwin, C. Avendano: Frequency-domain algorithms for audio signal enhancement based on transient modification, J. Audio Eng. Soc. **54**(9), 827–840 (2006)
12.67 A.V. Oppenheim, R.W. Schafer: *Discrete-Time Signal Processing* (Prentice-Hall, Englewood Cliffs 1989)
12.68 M. Goodwin, M. Wolters, R. Sridharan: Post-processing and computation in parametric and transform audio coders, AES 22nd Int. Conf.: Virtual, Synthetic, and Entertainment Audio (2002) pp. 149–158
12.69 S.F. Boll: Suppression of acoustic noise in speech using spectral subtraction, IEEE Trans. Acoust. Speech **27**(2), 113–120 (1979)
12.70 D.L. Donoho: De-noising by soft-thresholding, IEEE Trans. Inform. Theory **41**(3), 613–627 (1995)
12.71 S. Mallat, D. Donoho, A. Willsky: Best basis algorithm for signal enhancement, IEEE ICASSP, Vol. 3 (1995) pp. 1561–1564
12.72 J.M. Kates: Speech enhancement based on a sinusoidal model, J. Speech Hear. Res. **37**(2), 449–464 (1994)
12.73 T.F. Quatieri, R.J. McAulay: Speech transformations based on a sinusoidal representation, IEEE Trans. Acoust. Speech **34**(6), 1449–1464 (1986)
12.74 T.F. Quatieri, R.J. McAulay: Shape invariant time-scale and pitch modification of speech, IEEE Trans. Signal Process. **40**(3), 497–510 (1992)
12.75 D.L. Jones, T.W. Parks: Generation and combination of grains for music synthesis, Comput. Music J. **12**(2), 27–34 (1988)
12.76 C.A. Rodbro, M.G. Christensen, S.H. Jensen, S.V. Andersen: Compressed domain packet loss concealment of sinusoidally coded speech, IEEE ICASSP, Vol. 1 (2003) pp. 104–107
12.77 G. Wolberg: Recent advances in image morphing, Proc. of Comput. Graph. Int. (1996) pp. 64–71
12.78 M. Slaney, M. Covell, B. Lassiter: Automatic audio morphing, IEEE ICASSP, Vol. 2 (1996) pp. 1001–1004
12.79 H. Purnhagen, B. Edler, C. Ferekidis: Object-based analysis/synthesis audio coder for very low bit rates, 104th Audio Eng. Soc. Conv. (1998), Preprint 4747
12.80 M. Goodwin, C. Avendano: Parametric coding and frequency-domain processing for multichannel audio applications, AES 24th Int. Conf.: Multichannel Audio (2003) pp. 280–285
12.81 M.G. Christensen: Estimation and modeling problems in parametric audio coding. Ph.D. Thesis (Aalborg University, Aalborg 2005)
12.82 R.J. McAulay, T.F. Quatieri: Multirate sinusoidal transform coding at rates from 2.4 kbps to 8 kbps, IEEE ICASSP, Vol. 3 (1987) pp. 1645–1648
12.83 S. Ahmadi: New techniques for sinusoidal coding of speech at 2400 bps, Conf. Record of the Thirtieth Asilomar Conf. on Signals, Systems, and Comput., Vol. 1 (1996) pp. 770–774

6. Wiener and Adaptive Filters

J. Benesty, Y. Huang, J. Chen

The Wiener filter, named after its inventor, has been an extremely useful tool since its invention in the early 1930s. This optimal filter is not only popular in different aspects of speech processing but also in many other applications. This chapter presents the most fundamental results of the Wiener theory with an emphasis on the Wiener–Hopf equations, which are not convenient to solve in practice. An alternative approach to solving these equations directly is the use of an adaptive filter, which is why this work also describes the most classical adaptive algorithms that are able to converge, in a reasonable amount of time, to the optimal Wiener filter.

6.1 Overview

In his landmark manuscript on extrapolation, interpolation, and smoothing of stationary time series [6.1], Norbert Wiener was one of the first researchers to treat the filtering problem of estimating a process corrupted by additive noise. The optimum estimate that he derived, required the solution of an integral equation known as the Wiener–Hopf equation [6.2]. Soon after he published his work, Levinson formulated the same problem in discrete time [6.3]. Levinson's contribution has had a great impact on the field. Indeed, thanks to him, Wiener's ideas have become more accessible to many engineers. A very nice overview of linear filtering theory and the history of the different discoveries in this area can be found in [6.4].

In this chapter, we will show that the Wiener theory plays a fundamental role in system identification. For example, in many speech applications, impulse responses between loudspeakers (or speech sources) and microphones need to be identified. Thanks to many (adaptive) algorithms directly derived from the Wiener–Hopf equations, this task is now rather easy.

This chapter is organized as follows. Section 6.2 presents the four basic signal models used in this work. In Sect. 6.3, we derive the optimal Wiener filter for a single-input single-output (SISO) system. Section 6.4 explains what happens if the length of the modeling filter is shorter than the length of the true impulse response (this case always occurs in practice). It is extremely useful in many applications to be able to say how the input signal correlation matrix, which appears in the Wiener–Hopf equations, is conditioned. So we dedicate Sect. 6.5 to a detailed discussion on the condition number of this matrix. In Sect. 6.6, we present a collection of basic adaptive

filters. We insist on the classical normalized least-mean-square (NLMS) algorithm. Section 6.7 generalizes the Wiener filter to the multiple-input multiple-output (MIMO) system case. While this generalization is straightforward, the optimal solution does not always exist and identification problems may be possible only in some situations. Finally, we give our conclusions in Sect. 6.8.

6.2 Signal Models

In many speech applications, a system with a number of inputs and outputs needs to be identified. In this section, we explain the four basic signal models. This classification is now well accepted and is the basis of many interesting studies in different areas of control and signal processing.

6.2.1 SISO Model

The first model we consider is the single-input single-output (SISO) system, as shown in Fig. 6.1a. The output signal is given by

$$x(k) = h * s(k) + b(k) \,, \tag{6.1}$$

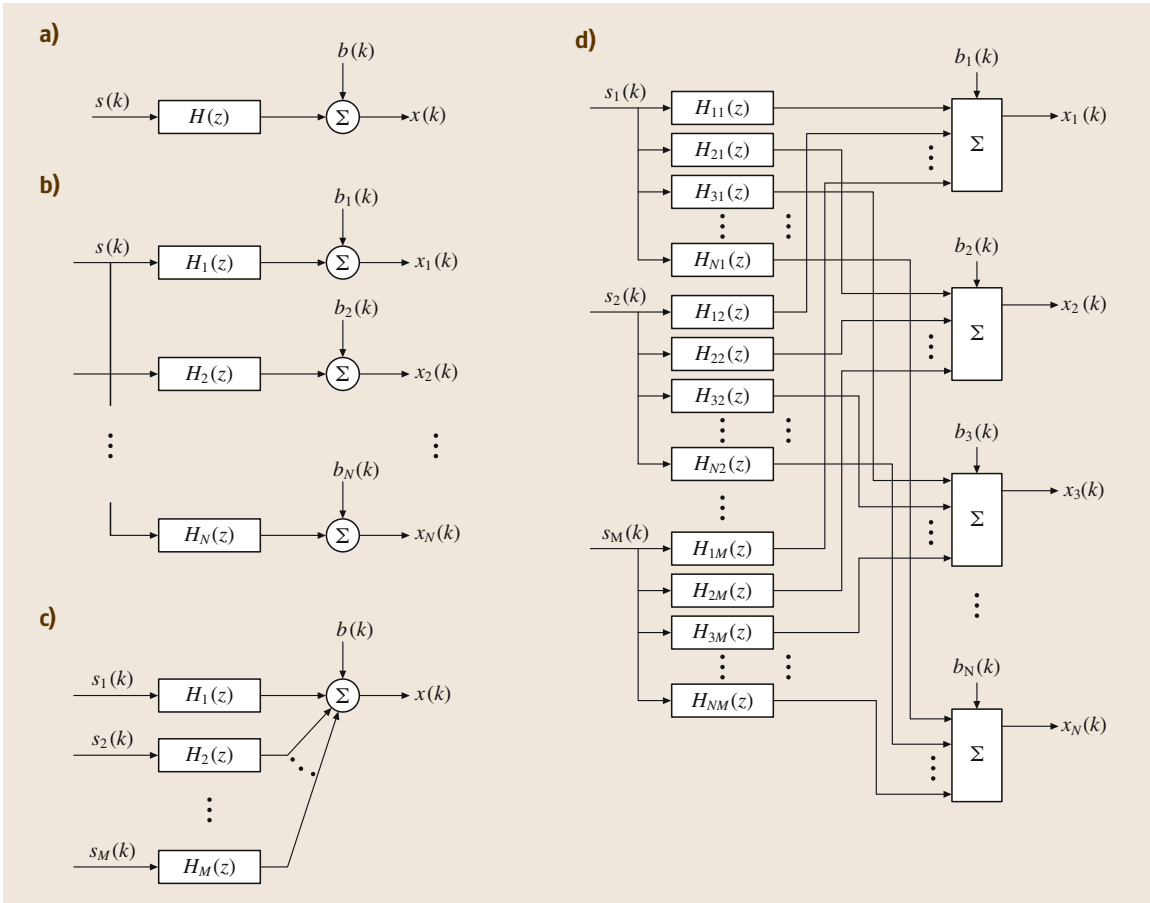


Fig. 6.1a–d Illustration of four distinct types of systems. **(a)** A single-input single-output (SISO) system. **(b)** A single-input multiple-output (SIMO) system. **(c)** A multiple-input single-output (MISO) system. **(d)** A multiple-input multiple-output (MIMO) system.

where h is the channel impulse response, the symbol $*$ denotes the linear convolution operator, $s(k)$ is the source signal at time k, and $b(k)$ is the additive noise at the output. Here we assume that the system is linear and shift-invariant. The channel impulse response is delineated usually with a finite impulse response (FIR) filter rather than an infinite impulse response (IIR) filter. In vector/matrix form, the SISO signal model (6.1) is written as:

$$x(k) = \boldsymbol{h}^{\mathrm{T}}\boldsymbol{s}(k) + b(k)\,, \tag{6.2}$$

where

$$\boldsymbol{h} = [h_0 \;\; h_1 \;\; \cdots \;\; h_{L-1}]^{\mathrm{T}}\,,$$

$$\boldsymbol{s}(k) = [s(k) \;\; s(k-1) \;\; \cdots \;\; s(k-L+1)]^{\mathrm{T}}\,,$$

where $[\cdot]^{\mathrm{T}}$ denotes the transpose of a matrix or a vector, and L is the channel length.

Using the z transform, the SISO signal model (6.2) is described as follows:

$$X(z) = H(z)S(z) + B(z)\,, \tag{6.3}$$

where $X(z)$, $S(z)$, and $B(z)$ are the z-transforms of $x(k)$, $s(k)$, and $b(k)$, respectively, and $H(z) = \sum_{l=0}^{L-1} h_l z^{-l}$.

The SISO model is simple and is probably the most widely used and studied model in communication, signal processing, and control theories.

6.2.2 SIMO Model

The diagram of a single-input multiple-output (SIMO) system is illustrated in Fig. 6.1b, in which there are N outputs from the same source as input and the n-th output is expressed as:

$$x_n(k) = \boldsymbol{h}_n^{\mathrm{T}}\boldsymbol{s}(k) + b_n(k)\,, \quad n = 1, 2, \cdots, N\,, \tag{6.4}$$

where $x_n(k)$, $\boldsymbol{h}_n$, and $b_n(k)$ are defined in a similar way to those in (6.2), and L is the length of the longest channel impulse response in this SIMO system. A more-comprehensive expression of the SIMO model is given by

$$\boldsymbol{x}(k) = \mathbf{H}\boldsymbol{s}(k) + \boldsymbol{b}(k)\,, \tag{6.5}$$

where

$$\boldsymbol{x}(k) = [x_1(k) \;\; x_2(k) \;\; \cdots \;\; x_N(k)]^{\mathrm{T}}\,,$$

$$\mathbf{H} = \begin{pmatrix} h_{1,0} & h_{1,1} & \cdots & h_{1,L-1} \\ h_{2,0} & h_{2,1} & \cdots & h_{2,L-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{N,0} & h_{N,1} & \cdots & h_{N,L-1} \end{pmatrix}_{N\times L}\,,$$

$$\boldsymbol{b}(k) = [b_1(k) \;\; b_2(k) \;\; \cdots \;\; b_N(k)]^{\mathrm{T}}\,.$$

The SIMO model (6.5) is described in the z-transform domain as:

$$\boldsymbol{X}(z) = \boldsymbol{H}(z)S(z) + \boldsymbol{B}(z)\,, \tag{6.6}$$

where

$$\boldsymbol{X}(z) = [X_1(z) \;\; X_2(z) \;\; \cdots \;\; X_N(z)]^{\mathrm{T}}\,,$$

$$\boldsymbol{H}(z) = [H_1(z) \;\; H_2(z) \;\; \cdots \;\; H_N(z)]^{\mathrm{T}}\,,$$

$$H_n(z) = \sum_{l=0}^{L-1} h_{n,l} z^{-l}, \quad n = 1, 2, \cdots, N\,,$$

$$\boldsymbol{B}(z) = [B_1(z) \;\; B_2(z) \;\; \cdots \;\; B_N(z)]^{\mathrm{T}}\,.$$

6.2.3 MISO Model

In the third type of systems as drawn in Fig. 6.1c, we suppose that there are M sources but only one output, whose signal is then expressed as:

$$\begin{aligned} x(k) &= \sum_{m=1}^{M} \boldsymbol{h}_m^{\mathrm{T}}\boldsymbol{s}_m(k) + b(k)\,, \\ &= \mathbf{h}^{\mathrm{T}}\mathbf{s}(k) + b(k)\,, \end{aligned} \tag{6.7}$$

where

$$\mathbf{h} = \left[\boldsymbol{h}_1^{\mathrm{T}} \;\; \boldsymbol{h}_2^{\mathrm{T}} \;\; \cdots \;\; \boldsymbol{h}_M^{\mathrm{T}}\right]^{\mathrm{T}}\,,$$

$$\boldsymbol{h}_m = [h_{m,0} \;\; h_{m,1} \;\; \cdots \;\; h_{m,L-1}]^{\mathrm{T}}\,,$$

$$\mathbf{s}(k) = \left[\boldsymbol{s}_1^{\mathrm{T}}(k) \;\; \boldsymbol{s}_2^{\mathrm{T}}(k) \;\; \cdots \;\; \boldsymbol{s}_M^{\mathrm{T}}(k)\right]^{\mathrm{T}}\,,$$

$$\boldsymbol{s}_m(k) = [s_m(k) \;\; s_m(k-1) \;\; \cdots \;\; s_m(k-L+1)]^{\mathrm{T}}\,.$$

In the z-transform domain, the MISO model is given by

$$X(z) = \boldsymbol{H}^{\mathrm{T}}(z)\boldsymbol{S}(z) + B(z)\,, \tag{6.8}$$

where

$$\boldsymbol{H}(z) = [H_1(z) \;\; H_2(z) \;\; \cdots \;\; H_M(z)]^{\mathrm{T}}\,,$$

$$H_m(z) = \sum_{l=0}^{L-1} h_{m,l} z^{-l}, \quad m = 1, 2, \cdots, M\,,$$

$$\boldsymbol{S}(z) = [S_1(z) \;\; S_2(z) \;\; \cdots \;\; S_M(z)]^{\mathrm{T}}\,.$$

Note that $\boldsymbol{H}(z)$ defined here is slightly different from that in (6.6). We do not deliberately distinguish them since their dimension can be easily deduced from the context if slight attention is paid.

6.2.4 MIMO Model

Figure 6.1d depicts a multiple-input multiple-output (MIMO) system. A MIMO system with M inputs and N outputs is referred to as an $M \times N$ system. At time k, we have

$$\boldsymbol{x}(k) = \mathbf{H}\boldsymbol{s}(k) + \boldsymbol{b}(k)\,, \tag{6.9}$$

where

$$\boldsymbol{x}(k) = [x_1(k)\ \ x_2(k)\ \ \cdots\ \ x_N(k)]^{\mathrm{T}}\,,$$

$$\mathbf{H} = (\boldsymbol{H}_1\ \ \boldsymbol{H}_2\ \ \cdots\ \ \boldsymbol{H}_M)\,,$$

$$\mathbf{H}_m = \begin{pmatrix} h_{1m,0} & h_{1m,1} & \cdots & h_{1m,L-1} \\ h_{2m,0} & h_{2m,1} & \cdots & h_{2m,L-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{Nm,0} & h_{Nm,1} & \cdots & h_{Nm,L-1} \end{pmatrix}_{N\times L}\,,$$

$$m = 1, 2, \cdots, M,$$

$$\boldsymbol{b}(k) = [b_1(k)\ \ b_2(k)\ \ \cdots\ \ b_N(k)]^{\mathrm{T}}\,,$$

where h_{nm} $(n = 1, 2, \cdots, N,\ m = 1, 2, \cdots, M)$ is the impulse response of the channel from input m to output n, and $\mathbf{s}(k)$ is defined similarly to that in (6.7). Again, we have the model presented in the z-transform domain as

$$\boldsymbol{X}(z) = \mathbf{H}(z)\boldsymbol{S}(z) + \boldsymbol{B}(z)\,, \tag{6.10}$$

where

$$\mathbf{H}(z) = \begin{pmatrix} H_{11}(z) & H_{12}(z) & \cdots & H_{1M}(z) \\ H_{21}(z) & H_{22}(z) & \cdots & H_{2M}(z) \\ \vdots & \vdots & \ddots & \vdots \\ H_{N1}(z) & H_{N2}(z) & \cdots & H_{NM}(z) \end{pmatrix}\,,$$

$$H_{nm}(z) = \sum_{l=0}^{L-1} h_{nm,l} z^{-l}\,, \quad n = 1, 2, \cdots, N\,,$$

$$m = 1, 2, \cdots, M\,.$$

Clearly the MIMO system is the most general model and the other three systems can be treated as special examples of a MIMO system.

6.3 Derivation of the Wiener Filter

In this section, we are interested in the SISO system represented by (6.2). We assume that $x(k)$ and the random noise signal $b(k)$ (independent of $s(k)$) are zero-mean and stationary.

With the Wiener theory, it is possible to identify the impulse response $\boldsymbol{h}$, given $s(k)$ and $x(k)$. Define the error signal,

$$\begin{aligned} e(k) &= x(k) - \hat{x}(k) \\ &= x(k) - \hat{\boldsymbol{h}}_{\mathrm{f}}^{\mathrm{T}} \boldsymbol{s}_{\mathrm{f}}(k)\,, \end{aligned} \tag{6.11}$$

where

$$\hat{\boldsymbol{h}}_{\mathrm{f}} = [\hat{h}_0\ \ \hat{h}_1\ \ \cdots\ \ \hat{h}_{L_{\mathrm{f}}-1}]^{\mathrm{T}}$$

is an estimate of $\boldsymbol{h}$ of length $L_{\mathrm{f}} \le L$ and

$$\boldsymbol{s}_{\mathrm{f}}(k) = [s(k)\ \ s(k-1)\ \ \cdots\ \ s(k-L_{\mathrm{f}}+1)]^{\mathrm{T}}\,.$$

To find the optimal filter, we need to minimize a cost function which is always built around the error signal (6.11). The usual choice for this criterion is the mean-square error (MSE) [6.5],

$$J(\hat{\boldsymbol{h}}_{\mathrm{f}}) = E\{e^2(k)\}\,, \tag{6.12}$$

where $E\{\cdot\}$ denotes mathematical expectation.

The optimal Wiener filter, $\hat{\boldsymbol{h}}_{\mathrm{f,o}}$, is the one that cancels the gradient of $J(\hat{\boldsymbol{h}}_{\mathrm{f}})$, i. e.,

$$\frac{\partial J(\hat{\boldsymbol{h}}_{\mathrm{f}})}{\partial \hat{\boldsymbol{h}}_{\mathrm{f}}} = \mathbf{0}_{L_{\mathrm{f}}\times 1}\,. \tag{6.13}$$

We have:

$$\begin{aligned} \frac{\partial J(\hat{\boldsymbol{h}}_{\mathrm{f}})}{\partial \hat{\boldsymbol{h}}_{\mathrm{f}}} &= 2E\left\{e(k)\frac{\partial e(k)}{\partial \hat{\boldsymbol{h}}_{\mathrm{f}}}\right\} \\ &= -2E\{e(k)\boldsymbol{s}_{\mathrm{f}}(k)\}\,. \end{aligned} \tag{6.14}$$

Therefore, at the optimum, we have:

$$E\{e_{\mathrm{o}}(k)\boldsymbol{s}_{\mathrm{f}}(k)\} = \mathbf{0}_{L_{\mathrm{f}}\times 1}\,, \tag{6.15}$$

where

$$e_{\mathrm{o}}(k) = x(k) - \hat{\boldsymbol{h}}_{\mathrm{f,o}}^{\mathrm{T}} \boldsymbol{s}_{\mathrm{f}}(k) \tag{6.16}$$

is the error signal for which $J(\hat{\boldsymbol{h}}_{\mathrm{f}})$ is minimized (i. e., the optimal filter). Expression (6.15) is called the *principle of orthogonality*.

The optimal estimate of $x(k)$ is:

$$\hat{x}_{\mathrm{o}}(k) = \hat{\boldsymbol{h}}_{\mathrm{f,o}}^{\mathrm{T}} \boldsymbol{s}_{\mathrm{f}}(k)\,. \tag{6.17}$$

It is then easy to check, with the help of the principle of orthogonality, that we also have:

$$E\left\{e_{\mathrm{o}}(k)\hat{x}_{\mathrm{o}}(k)\right\} = 0\,. \tag{6.18}$$

The previous expression is called the *corollary to the principle of orthogonality*.

If we substitute (6.16) into (6.15), we find the *Wiener–Hopf equations*,

$$\mathbf{R}_\mathrm{f}\hat{\boldsymbol{h}}_{\mathrm{f,o}} = \boldsymbol{p}_\mathrm{f}\,, \tag{6.19}$$

where

$$\mathbf{R}_\mathrm{f} = E\left\{\boldsymbol{s}_\mathrm{f}(k)\boldsymbol{s}_\mathrm{f}^\mathrm{T}(k)\right\}$$

is the correlation matrix of the signal $s(k)$ and

$$\boldsymbol{p}_\mathrm{f} = E\{\boldsymbol{s}_\mathrm{f}(k)x(k)\}$$

is the cross-correlation vector between $\boldsymbol{s}_\mathrm{f}(k)$ and $x(k)$.

The correlation matrix is symmetric and positive semidefinite. It is also Toeplitz, i. e., a matrix which has constant values along diagonals,

$$\mathbf{R}_\mathrm{f} = \begin{pmatrix} r(0) & r(1) & \cdots & r(L_\mathrm{f}-1) \\ r(1) & r(0) & \cdots & r(L_\mathrm{f}-2) \\ \vdots & \vdots & \ddots & \vdots \\ r(L_\mathrm{f}-1) & r(L_\mathrm{f}-2) & \cdots & r(0) \end{pmatrix},$$

with $r(l) = E\{s(k)s(k-l)\}$, $l = 0, 1, \cdots, L_\mathrm{f}-1$. In the SISO system case, this matrix is usually positive definite even for quasistationary signals like speech; however, it can be very ill-conditioned.

Assuming that $\mathbf{R}_\mathrm{f}$ is nonsingular, the optimal Wiener filter is:

$$\hat{\boldsymbol{h}}_{\mathrm{f,o}} = \mathbf{R}_\mathrm{f}^{-1}\boldsymbol{p}_\mathrm{f}\,. \tag{6.20}$$

The MSE can be rewritten as:

$$J(\hat{\boldsymbol{h}}_\mathrm{f}) = \sigma_x^2 - 2\boldsymbol{p}_\mathrm{f}^\mathrm{T}\hat{\boldsymbol{h}}_\mathrm{f} + \hat{\boldsymbol{h}}_\mathrm{f}^\mathrm{T}\mathbf{R}_\mathrm{f}\hat{\boldsymbol{h}}_\mathrm{f}\,, \tag{6.21}$$

where $\sigma_x^2 = E\{x^2(k)\}$ is the variance of the input signal $x(k)$. The criterion $J(\hat{\boldsymbol{h}}_\mathrm{f})$ is a quadratic function of the filter coefficient vector $\hat{\boldsymbol{h}}_\mathrm{f}$ and has a single minimum point. This point combines the optimal Wiener filter, as shown above, and a value called the minimum MSE (MMSE), which is obtained by substituting (6.20) into (6.21):

$$\begin{aligned} J_\mathrm{min} &= J(\hat{\boldsymbol{h}}_{\mathrm{f,o}}) \\ &= \sigma_x^2 - \boldsymbol{p}_\mathrm{f}^\mathrm{T}\mathbf{R}_\mathrm{f}^{-1}\boldsymbol{p}_\mathrm{f} \\ &= \sigma_x^2 - \sigma_{\hat{x}_\mathrm{o}}^2\,, \end{aligned} \tag{6.22}$$

where $\sigma_{\hat{x}_\mathrm{o}}^2 = E\{\hat{x}_\mathrm{o}^2(k)\}$ is the variance of the optimal filter output signal $\hat{x}_\mathrm{o}(k)$. This MMSE can be rewritten as:

$$J_\mathrm{min} = \sigma_b^2 + \boldsymbol{h}^\mathrm{T}\mathbf{R}\boldsymbol{h} - \hat{\boldsymbol{h}}_{\mathrm{f,o}}^\mathrm{T}\mathbf{R}_\mathrm{f}\hat{\boldsymbol{h}}_{\mathrm{f,o}}\,, \tag{6.23}$$

where $\sigma_b^2 = E\{b^2(k)\}$ is the variance of the noise and $\mathbf{R} = E\{\boldsymbol{s}(k)\boldsymbol{s}^\mathrm{T}(k)\}$. The value J_min is bounded,

$$\sigma_b^2 \le J_\mathrm{min} \le \sigma_b^2 + \boldsymbol{h}^\mathrm{T}\mathbf{R}\boldsymbol{h}\,, \quad \forall L_\mathrm{f}\,. \tag{6.24}$$

We can easily check that for $L_\mathrm{f} = L$, $J_\mathrm{min} = \sigma_b^2$, and as L_f decreases compared to L, J_min gets closer to its maximum value $\sigma_b^2 + \boldsymbol{h}^\mathrm{T}\mathbf{R}\boldsymbol{h}$.

We define the normalized MMSE as:

$$\tilde{J}_\mathrm{min} = \frac{J_\mathrm{min}}{\sigma_x^2} = 1 - \frac{\sigma_{\hat{x}_\mathrm{o}}^2}{\sigma_x^2}\,. \tag{6.25}$$

According to (6.24), the normalized MMSE always satisfies,

$$\frac{\sigma_b^2}{\sigma_x^2} \le \tilde{J}_\mathrm{min} \le 1\,. \tag{6.26}$$

6.4 Impulse Response Tail Effect

In many scenarios, the impulse response that we try to estimate is either very long or its length is not known so that the length (L_f) of any FIR modeling filter $\hat{\boldsymbol{h}}_\mathrm{f}$ will usually be shorter than the length (L) of the actual impulse response. Let us split this impulse response into two parts:

$$\mathbf{h} = \begin{pmatrix} \boldsymbol{h}_\mathrm{f} \\ \boldsymbol{h}_\mathrm{t} \end{pmatrix}$$

where $\boldsymbol{h}_\mathrm{f}$ is a vector of size L_f and $\boldsymbol{h}_\mathrm{t}$ is the *tail* of the impulse response that is not modeled by $\hat{\boldsymbol{h}}_\mathrm{f}$. Equation (6.2), which represents the SISO system, is now:

$$x(k) = \boldsymbol{h}_\mathrm{f}^\mathrm{T}\boldsymbol{s}_\mathrm{f}(k) + \boldsymbol{h}_\mathrm{t}^\mathrm{T}\boldsymbol{s}_\mathrm{t}(k-L_\mathrm{f}) + b(k)\,, \tag{6.27}$$

where

$$\begin{aligned} \boldsymbol{s}_\mathrm{t}(k-L_\mathrm{f}) = [&s(k-L_\mathrm{f})\;\; s(k-L_\mathrm{f}-1) \\ &\cdots\;\; s(k-L+1)]^\mathrm{T}\,. \end{aligned}$$

Substituting (6.27) into the cross-correlation vector, we obtain,

$$\boldsymbol{p}_\mathrm{f} = E\{\boldsymbol{s}_\mathrm{f}(k)x(k)\} = \mathbf{R}_\mathrm{f}\boldsymbol{h}_\mathrm{f} + \mathbf{R}_\mathrm{t}(L_\mathrm{f})\boldsymbol{h}_\mathrm{t}\,, \tag{6.28}$$

with $\mathbf{R}_\mathrm{t}(L_\mathrm{f}) = E\{\boldsymbol{s}_\mathrm{f}(k)\boldsymbol{s}_\mathrm{t}^\mathrm{T}(k-L_\mathrm{f})\}$. Finally, inserting the previous expression into the Wiener–Hopf equations (6.20), we obtain:

$$\hat{\boldsymbol{h}}_{\mathrm{f,o}} = \boldsymbol{h}_\mathrm{f} + \mathbf{R}_\mathrm{f}^{-1}\mathbf{R}_\mathrm{t}(L_\mathrm{f})\boldsymbol{h}_\mathrm{t}\,. \tag{6.29}$$

It is clear from (6.29) that the underestimation of the length of the impulse response in the Wiener method

will introduce a bias [equal to $\mathbf{R}_\mathrm{f}^{-1}\mathbf{R}_\mathrm{t}(L_\mathrm{f})\boldsymbol{h}_\mathrm{t}$] in the coefficients of the optimal filter. This bias depends on two things: the energy of the tail impulse response and the correlation of the input signal $s(k)$. If $s(k)$ is white, there is no bias since in this particular case the matrix $\mathbf{R}_\mathrm{t}(L_\mathrm{f})$ is zero. But for highly correlated signals like speech, $\mathbf{R}_\mathrm{t}(L_\mathrm{f})$ may not be negligible and the second term on the right-hand side of (6.29) may therefore be amplified if the energy of the tail is significative. As a consequence, it is important in practice to have a rough idea of the physics of the system, in order to choose an appropriate length for the modeling filter for good identification. As we can see, increasing the length of the filter will improve the accuracy of the solution. On the other hand, the complexity for solving the linear system will increase and the conditioning of $\mathbf{R}_\mathrm{f}$ will be worsened. Therefore, depending on the application, a reasonable balance has to be found.

For simplification, in the rest of this chapter, we will assume that $L_\mathrm{f} = L$ so that we can drop the subscript 'f' in all variables. In this scenario: $\mathbf{R}_\mathrm{f} = \mathbf{R}$, $\boldsymbol{s}_\mathrm{f}(k) = \boldsymbol{s}(k)$, $\hat{\boldsymbol{h}}_\mathrm{f,o} = \hat{\boldsymbol{h}}_\mathrm{o}$, etc.

6.5 Condition Number

The correlation matrix that appears in the Wiener–Hopf equations needs to be inverted to find the optimal filter. If this matrix is ill-conditioned and the data is perturbed, the accuracy of the solution will suffer a lot if the linear system is solved directly. One way to improve the accuracy is to regularize the covariance matrix. However, this regularization depends on the condition number: the higher the condition number, the larger the regularization. So it is important to be able to estimate this condition number in an efficient way, in order to use this information to improve the quality of the solution. Many other problems require the knowledge of this condition number for different reasons. For example, the performance of many adaptive algorithms depends on this number. Therefore, it is of great interest to have a detailed discussion of this topic here and to develop a practical algorithm to determine this condition number.

6.5.1 Decomposition of the Correlation Matrix

For a vector of length $L+1$,

$$\boldsymbol{s}_{L+1}(k) = [s(k)\ \ s(k-1)\ \ \cdots\ \ s(k-L)]^\mathrm{T}\,,$$

the covariance matrix of size $(L+1)\times(L+1)$ is:

$$\begin{aligned}\mathbf{R}_{L+1} &= E\left\{\boldsymbol{s}_{L+1}(k)\boldsymbol{s}_{L+1}^\mathrm{T}(k)\right\}\\ &= \begin{pmatrix} r(0) & \boldsymbol{r}_L^\mathrm{T} \\ \boldsymbol{r}_L & \mathbf{R}_L \end{pmatrix} = \begin{pmatrix} \mathbf{R}_L & \boldsymbol{r}_{\mathrm{b},L} \\ \boldsymbol{r}_{\mathrm{b},L}^\mathrm{T} & r(0) \end{pmatrix}\,, \end{aligned} \tag{6.30}$$

where $\mathbf{R}_L = E\{\boldsymbol{s}(k)\boldsymbol{s}^\mathrm{T}(k)\}$ and

$$\begin{aligned}\boldsymbol{r}_L &= [r(1)\ \ r(2)\ \ \cdots\ \ r(L)]^\mathrm{T}\,,\\ \boldsymbol{r}_{\mathrm{b},L} &= [r(L)\ \ r(L-1)\ \ \cdots\ \ r(1)]^\mathrm{T}\,.\end{aligned}$$

By using the Schur complements, it is easy to invert $\mathbf{R}_{L+1}$:

$$\mathbf{R}_{L+1}^{-1} = \begin{pmatrix} \mathbf{R}_L^{-1} + \varrho_L^{-1}\boldsymbol{b}_L\boldsymbol{b}_L^\mathrm{T} & -\varrho_L^{-1}\boldsymbol{b}_L \\ -\varrho_L^{-1}\boldsymbol{b}_L^\mathrm{T} & \varrho_L^{-1} \end{pmatrix}\,, \tag{6.31}$$

where

$$\begin{aligned}\boldsymbol{b}_L &= \mathbf{R}_L^{-1}\boldsymbol{r}_{\mathrm{b},L}\\ &= [b_{L,1}\ \ b_{L,2}\ \ \cdots\ \ b_{L,L}]^\mathrm{T}\end{aligned} \tag{6.32}$$

is the backward predictor of length L,

$$\begin{aligned}\varrho_L &= r(0) - \boldsymbol{r}_{\mathrm{b},L}^\mathrm{T}\boldsymbol{b}_L\\ &= r(0) - \boldsymbol{r}_L^\mathrm{T}\boldsymbol{a}_L\end{aligned} \tag{6.33}$$

is the prediction error energy, and $\boldsymbol{a}_L = \mathbf{J}_L\boldsymbol{b}_L$ is the forward predictor with $\mathbf{J}_L$ being the co-identity matrix. Equation (6.31) is important and will be used later for a fast computation of the condition number.

6.5.2 Condition Number with the Frobenius Norm

Usually, the condition number is computed by using the 2-norm matrix. However, in the context of Toeplitz matrices, it is more convenient to use the Frobenius norm as explained below and in [6.6, 7].

To simplify the notation, in this subsection we take $\mathbf{R}_{L+1} = \mathbf{R}$. This matrix is symmetric, positive, and assumed to be nonsingular. It can be diagonalized as follows:

$$\mathbf{Q}^\mathrm{T}\mathbf{R}\mathbf{Q} = \boldsymbol{\Lambda}\,, \tag{6.34}$$

where

$$\mathbf{Q}^\mathrm{T}\mathbf{Q} = \mathbf{Q}\mathbf{Q}^\mathrm{T} = \mathbf{I}\,, \tag{6.35}$$

$$\boldsymbol{\Lambda} = \mathrm{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_{L+1}\}\,, \tag{6.36}$$

and $0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_{L+1}$. By definition, the square root of $\mathbf{R}$ is:

$$\mathbf{R}^{1/2} = \mathbf{Q}\boldsymbol{\Lambda}^{1/2}\mathbf{Q}^{\mathrm{T}} . \tag{6.37}$$

The condition number of a matrix $\mathbf{R}$ is [6.8]:

$$\chi(\mathbf{R}) = \|\mathbf{R}\|\|\mathbf{R}^{-1}\| , \tag{6.38}$$

where $\|\cdot\|$ can be any matrix norm. Note that $\chi(\mathbf{R})$ depends on the underlying norm and subscripts will be used to distinguish the different condition numbers.

Consider the Frobenius norm:

$$\|\mathbf{R}\|_{\mathrm{F}} = \{\mathrm{tr}(\mathbf{R}^{\mathrm{T}}\mathbf{R})\}^{1/2} . \tag{6.39}$$

We can easily check that, indeed, $\|\cdot\|_{\mathrm{F}}$ is a matrix norm since for any real matrices $\mathbf{A}$ and $\mathbf{B}$ and a real scalar c, the following three conditions are satisfied:

- $\|\mathbf{A}\|_{\mathrm{F}} \geq 0$ and $\|\mathbf{A}\|_{\mathrm{F}} = 0$ if $\mathbf{A} = \mathbf{0}_{(L+1)\times(L+1)}$,
- $\|\mathbf{A}+\boldsymbol{B}\|_{\mathrm{F}} \leq \|\mathbf{A}\|_{\mathrm{F}} + \|\boldsymbol{B}\|_{\mathrm{F}}$,
- $\|c\mathbf{A}\|_{\mathrm{F}} = |c|\|\mathbf{A}\|_{\mathrm{F}}$.

We have:

$$\|\mathbf{R}^{1/2}\|_{\mathrm{F}} = \{\mathrm{tr}(\mathbf{R})\}^{1/2} = \left\{\sum_{l=1}^{L+1}\lambda_l\right\}^{1/2} \tag{6.40}$$

and

$$\|\mathbf{R}^{-1/2}\|_{\mathrm{F}} = \{\mathrm{tr}(\mathbf{R}^{-1})\}^{1/2} = \left\{\sum_{l=1}^{L+1}\frac{1}{\lambda_l}\right\}^{1/2} . \tag{6.41}$$

Hence, the condition number of $\mathbf{R}^{1/2}$ associated with $\|\cdot\|_{\mathrm{F}}$ is:

$$\chi_{\mathrm{F}}(\mathbf{R}^{1/2}) = \|\mathbf{R}^{1/2}\|_{\mathrm{F}}\|\mathbf{R}^{-1/2}\|_{\mathrm{F}} \geq L+1 . \tag{6.42}$$

(The inequality in the previous expression is easy to show by using the Cauchy–Schwartz inequality.) In this section, we choose to work on $\chi_{\mathrm{F}}(\mathbf{R}^{1/2})$ [rather than $\chi_{\mathrm{F}}(\mathbf{R})$], because efficient algorithms can be derived to estimate its value, as will be shown in the next subsection. As far as we know, it does not seem obvious how to estimate $\chi_{\mathrm{F}}(\mathbf{R})$ efficiently.

If $\chi(\mathbf{R}^{1/2})$ is large, then $\mathbf{R}^{1/2}$ is said to be an ill-conditioned matrix. Note that this is a norm-dependent property. However, according to [6.8], any two condition numbers $\chi_\alpha(\mathbf{R}^{1/2})$ and $\chi_\beta(\mathbf{R}^{1/2})$ are equivalent in that constants c_1 and c_2 can be found for which:

$$c_1\chi_\alpha(\mathbf{R}^{1/2}) \leq \chi_\beta(\mathbf{R}^{1/2}) \leq c_2\chi_\alpha(\mathbf{R}^{1/2}) . \tag{6.43}$$

For example, for the 1- and 2-norm matrices and for $\mathbf{R}$, we can show [6.8]

$$\frac{1}{(L+1)^2}\chi_2(\mathbf{R}) \leq \frac{1}{L+1}\chi_1(\mathbf{R}) \leq \chi_2(\mathbf{R}) . \tag{6.44}$$

We now show the same principle for the F- and 2-norm matrices and for $\mathbf{R}^{1/2}$. We recall that:

$$\chi_2(\mathbf{R}^{1/2}) = \sqrt{\frac{\lambda_{L+1}}{\lambda_1}} . \tag{6.45}$$

Since $\mathrm{tr}(\mathbf{R}^{-1}) \geq 1/\lambda_1$ and $\mathrm{tr}(\mathbf{R}) \geq \lambda_{L+1}$, we have

$$\mathrm{tr}(\mathbf{R})\mathrm{tr}(\mathbf{R}^{-1}) \geq \frac{\mathrm{tr}(\mathbf{R})}{\lambda_1} \geq \frac{\lambda_{L+1}}{\lambda_1} , \tag{6.46}$$

hence,

$$\chi_{\mathrm{F}}(\mathbf{R}^{1/2}) \geq \chi_2(\mathbf{R}^{1/2}) . \tag{6.47}$$

Also, since $\mathrm{tr}(\mathbf{R}) \leq (L+1)\lambda_{L+1}$ and $\mathrm{tr}(\mathbf{R}^{-1}) \leq (L+1)/\lambda_1$, we obtain:

$$\mathrm{tr}(\mathbf{R})\mathrm{tr}(\mathbf{R}^{-1}) \leq (L+1)\frac{\mathrm{tr}(\mathbf{R})}{\lambda_1} \leq (L+1)^2\frac{\lambda_{L+1}}{\lambda_1} , \tag{6.48}$$

thus,

$$\chi_{\mathrm{F}}(\mathbf{R}^{1/2}) \leq (L+1)\chi_2(\mathbf{R}^{1/2}) . \tag{6.49}$$

Therefore, we deduce that

$$\chi_2(\mathbf{R}^{1/2}) \leq \chi_{\mathrm{F}}(\mathbf{R}^{1/2}) \leq (L+1)\chi_2(\mathbf{R}^{1/2}) . \tag{6.50}$$

Moreover, by using the two inequalities,

$$\left(\sum_{l=1}^{L+1}\beta_l\right)^2 \geq \sum_{l=1}^{L+1}\beta_l^2 , \tag{6.51}$$

$$\left(\sum_{l=1}^{L+1}\beta_l\right)^2 \leq (L+1)\sum_{l=1}^{L+1}\beta_l^2 , \tag{6.52}$$

where $\beta_l > 0, \forall l$, it is easy to show that

$$\frac{1}{L+1}\chi_{\mathrm{F}}^2(\mathbf{R}^{1/2}) \leq \chi_{\mathrm{F}}(\mathbf{R}) \leq \chi_{\mathrm{F}}^2(\mathbf{R}^{1/2}) \leq (L+1)\chi_{\mathrm{F}}(\mathbf{R}) . \tag{6.53}$$

Note that $\chi_2(\mathbf{R}) = \chi_2^2(\mathbf{R}^{1/2})$ but $\chi_{\mathrm{F}}(\mathbf{R}) \neq \chi_{\mathrm{F}}^2(\mathbf{R}^{1/2})$. According to expressions (6.50) and (6.53), $\chi_{\mathrm{F}}(\mathbf{R}^{1/2})$ and $\chi_{\mathrm{F}}^2(\mathbf{R}^{1/2})$ are a good measure of the condition number of matrices $\mathbf{R}^{1/2}$ and $\mathbf{R}$, respectively. Basically, there is no difference in the trend of the condition numbers of $\mathbf{R}$ and $\mathbf{R}^{1/2}$. In other words, if $\mathbf{R}^{1/2}$ is ill-conditioned (resp. well-conditioned) so is $\mathbf{R}$. In the next subsection, we will show how to compute $\chi_{\mathrm{F}}^2(\mathbf{R}^{1/2})$ by using the Levinson–Durbin algorithm.

Table 6.1 Computation of the condition number with the Levinson–Durbin algorithm

Initialization:	$\varrho_0 = r(0)$
Levinson-Durbin algorithm:	$k_l = \frac{1}{\varrho_{l-1}}\left[r(l) - \boldsymbol{a}_{l-1}^{\mathrm{T}} \boldsymbol{J}_{l-1} \boldsymbol{r}_{l-1}\right]$ $\boldsymbol{a}_l = \begin{pmatrix} \boldsymbol{a}_{l-1} \\ 0 \end{pmatrix} - k_l \boldsymbol{J}_l \begin{pmatrix} -1 \\ \boldsymbol{a}_{l-1} \end{pmatrix}$ $\varrho_l = \varrho_{l-1}(1-k_l^2)$ $l = 1, 2, \cdots, L$
Condition number:	$\chi_{\mathrm{F}}^2\left(\mathbf{R}_{L+1}^{1/2}\right) = (L+1)r(0)\sum_{l=0}^{L} \varrho_l^{-1}\left[1 + \boldsymbol{a}_l^{\mathrm{T}} \boldsymbol{a}_l\right]$

6.5.3 Fast Computation of the Condition Number

In this subsection, we need to compute the two norms $\left\|\mathbf{R}_{L+1}^{1/2}\right\|_{\mathrm{F}}^2$ and $\left\|\mathbf{R}_{L+1}^{-1/2}\right\|_{\mathrm{F}}^2$ efficiently. The calculation of the first is straightforward. Indeed:

$$\left\|\mathbf{R}_{L+1}^{1/2}\right\|_{\mathrm{F}}^2 = \mathrm{tr}(\mathbf{R}_{L+1}) = (L+1)r(0)\,. \tag{6.54}$$

Expression (6.54) requires one multiplication only. Consider the matrix $\mathbf{G}_{L+1} = \mathbf{R}_{L+1}^{-1}$ where its diagonal elements are $g_{L+1,ii}$, $i = 1, 2, \cdots, L+1$. It is clear from (6.31) that the last diagonal component of $\mathbf{G}_{L+1}$ is $g_{L+1,(L+1)(L+1)} = \varrho_L^{-1}$. The L-th diagonal element of $\mathbf{G}_{L+1}$ is $g_{L+1,\mathrm{LL}} = \varrho_{L-1}^{-1} + \varrho_L^{-1} b_{L,L}^2$. Continuing the same process, we easily find:

$$g_{L+1,ii} = \varrho_{i-1}^{-1} + \sum_{l=i}^{L} \varrho_l^{-1} b_{l,i}^2\,, \tag{6.55}$$

with $\varrho_0 = r(0)$. Therefore, from (6.55) we deduce that:

$$\begin{aligned}\left\|\mathbf{R}_{L+1}^{-1/2}\right\|_{\mathrm{F}}^2 &= \mathrm{tr}(\mathbf{G}_{L+1}) \\ &= \sum_{l=0}^{L} \varrho_l^{-1}\left(1 + \boldsymbol{b}_l^{\mathrm{T}} \boldsymbol{b}_l\right) \\ &= \sum_{l=0}^{L} \varrho_l^{-1}\left(1 + \boldsymbol{a}_l^{\mathrm{T}} \boldsymbol{a}_l\right)\,, \end{aligned} \tag{6.56}$$

with $\boldsymbol{a}_0^{\mathrm{T}} \boldsymbol{a}_0 = \boldsymbol{b}_0^{\mathrm{T}} \boldsymbol{b}_0 = 0$.

Finally, the condition number is:

$$\chi_{\mathrm{F}}^2\left(\mathbf{R}_{L+1}^{1/2}\right) = (L+1)r(0)\sum_{l=0}^{L} \varrho_l^{-1}\left(1 + \boldsymbol{a}_l^{\mathrm{T}} \boldsymbol{a}_l\right)\,. \tag{6.57}$$

By using the Toeplitz structure, the Levinson–Durbin algorithm solves the linear prediction equation, $\boldsymbol{a}_L = \mathbf{R}_L^{-1} \boldsymbol{r}_L$, in $O(L^2)$ operations instead of $O(L^3)$. This algorithm computes all predictors $\boldsymbol{a}_l$, $l = 1, 2, \cdots, L$, and this is exactly what we need to compute (6.57). Expression (6.57) also shows a very nice link between the condition number and the predictors of all orders. This algorithm, which has roughly the same complexity as the Levinson–Durbin algorithm, is summarized in Table 6.1. Note that a very efficient algorithm was recently proposed by *Dias* and *Leitão* [6.9] to compute $\mathrm{tr}\{\mathbf{T}\mathbf{R}^{-1}\}$ (where $\mathbf{T}$ is a Toeplitz matrix, this form is a much more-general form than the one used in this section) with the Trench algorithm. Using these techniques here, we can further reduce the complexity [to $\mathcal{O}(L \ln L)$] for the estimation of the overall algorithm.

6.6 Adaptive Algorithms

Solving the Wiener–Hopf equations directly is not very practical, so adaptive algorithms are usually preferred to find the optimal Wiener filter. The aim of this section is to present a couple of basic adaptive algorithms that converge to the actual impulse response $\boldsymbol{h}$ and where the inversion of the correlation matrix $\mathbf{R}$ is avoided.

6.6.1 Deterministic Algorithm

The deterministic or steepest-descent algorithm is actually an iterative algorithm. It is summarized by the simple recursion,

$$\begin{aligned}\hat{\boldsymbol{h}}(k) &= \hat{\boldsymbol{h}}(k-1) + \mu[\boldsymbol{p} - \mathbf{R}\hat{\boldsymbol{h}}(k-1)]\,, \\ & \qquad k = 0, 1, 2, \cdots\,, \end{aligned} \tag{6.58}$$

where μ is a positive constant called the step-size parameter. In this algorithm, $\boldsymbol{p}$ and $\mathbf{R}$ are supposed to be known. The deterministic algorithm can be reformulated with the error signal:

$$e(k) = x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)\,, \tag{6.59}$$

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu E\{\boldsymbol{s}(k)e(k)\}\,. \tag{6.60}$$

Now the important question is: what are the conditions on μ to make the algorithm converge to the true impulse response $\boldsymbol{h}$? To answer this question, we will examine the *natural modes* of the algorithm [6.10].

We define the *misalignment vector* as,

$$\boldsymbol{m}(k) = \boldsymbol{h} - \hat{\boldsymbol{h}}(k)\,, \tag{6.61}$$

which is the difference between the true impulse response and the estimated one at time k. The positive quantity $\|\boldsymbol{m}(k)\|_2^2/\|\boldsymbol{h}\|_2^2$ is called the *normalized misalignment*. If we substitute (6.2) into the cross-correlation vector, we get,

$$\boldsymbol{p} = E\{\boldsymbol{s}(k)x(k)\} = \mathbf{R}\boldsymbol{h}\,. \tag{6.62}$$

Inserting (6.62) into (6.58) and subtracting $\boldsymbol{h}$ on both sides of the equation, we obtain:

$$\boldsymbol{m}(k) = (\mathbf{I} - \mu\mathbf{R})\boldsymbol{m}(k-1)\,, \tag{6.63}$$

where $\mathbf{I}$ is the identity matrix. Using the eigendecomposition of $\mathbf{R} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^{\mathrm{T}}$ in the previous expression, we get the equivalent form,

$$\boldsymbol{v}(k) = (\mathbf{I} - \mu\boldsymbol{\Lambda})\boldsymbol{v}(k-1)\,, \tag{6.64}$$

where

$$\boldsymbol{v}(k) = \mathbf{Q}^{\mathrm{T}}\boldsymbol{m}(k) = \mathbf{Q}^{\mathrm{T}}[\boldsymbol{h} - \hat{\boldsymbol{h}}(k)]\,. \tag{6.65}$$

Thus, for the l-th natural mode of the steepest-descent algorithm, we have [6.5]

$$v_l(k) = (1 - \mu\lambda_l)v_l(k-1)\,, \quad l = 1, 2, \cdots, L \tag{6.66}$$

or, equivalently,

$$v_l(k) = (1 - \mu\lambda_l)^k v_l(0)\,, \quad l = 1, 2, \cdots, L\,. \tag{6.67}$$

The algorithm converges if,

$$\lim_{k\to\infty} v_l(k) = 0\,, \quad \forall l\,. \tag{6.68}$$

In this case,

$$\lim_{k\to\infty} \hat{\boldsymbol{h}}(k) = \boldsymbol{h}\,. \tag{6.69}$$

It is straightforward to see from (6.67) that a necessary and sufficient condition for the stability of the deterministic algorithm is that,

$$-1 < 1 - \mu\lambda_l < 1\,, \quad \forall l\,, \tag{6.70}$$

which implies,

$$0 < \mu < \frac{2}{\lambda_l}\,, \quad \forall l\,, \tag{6.71}$$

or

$$0 < \mu < \frac{2}{\lambda_{\max}}\,, \tag{6.72}$$

where $\lambda_{\max}$ is the largest eigenvalue of the correlation matrix $\mathbf{R}$.

Let us evaluate the time needed for each natural mode to converge to a given value. Expression (6.67) gives:

$$\ln\frac{|v_l(k)|}{|v_l(0)|} = k\ln|1 - \mu\lambda_l|\,, \tag{6.73}$$

hence,

$$k = \frac{1}{\ln|1 - \mu\lambda_l|}\ln\frac{|v_l(k)|}{|v_l(0)|}\,. \tag{6.74}$$

The *time constant*, τ_l, for the l-th natural mode is defined by taking $|v_l(k)|/|v_l(0)| = 1/e$ (where e is the base of the natural logarithm) in (6.74). Therefore,

$$\tau_l = \frac{-1}{\ln|1 - \mu\lambda_l|}\,. \tag{6.75}$$

We can link the time constant with the condition number of the correlation matrix $\mathbf{R}$. First, let

$$\mu = \frac{\alpha}{\lambda_{\max}}\,, \tag{6.76}$$

where

$$0 < \alpha < 2\,, \tag{6.77}$$

to guaranty the convergence of the algorithm. α is called the normalized step-size parameter. Suppose that the smallest eigenvalue is $\lambda_1 = \lambda_{\min}$; in this case,

$$\tau_1 = \frac{-1}{\ln|1 - \alpha\lambda_{\min}/\lambda_{\max}|} = \frac{-1}{\ln|1 - \alpha/\chi_2(\mathbf{R})|}\,, \tag{6.78}$$

where $\chi_2(\mathbf{R}) = \lambda_{\max}/\lambda_{\min}$. We see that the convergence time of the slowest natural mode depends on the conditioning of $\mathbf{R}$.

From (6.65), we deduce that,

$$\begin{aligned}\boldsymbol{m}^{\mathrm{T}}(k)\boldsymbol{m}(k) &= \boldsymbol{v}^{\mathrm{T}}(k)\boldsymbol{v}(k)\\ &= \left\|\boldsymbol{h}-\hat{\boldsymbol{h}}(k)\right\|_2^2\\ &= \sum_{l=1}^{L}\lambda_l(1-\mu\lambda_l)^k v_l(0)\,. \end{aligned} \tag{6.79}$$

This value gives an idea on the global convergence of the filter to the true impulse response. This convergence is clearly governed by the smallest eigenvalues of $\mathbf{R}$.

We now examine the transient behavior of the MSE. Using (6.2), the error signal (6.59) can be rewritten as

$$\begin{aligned}e(k) &= x(k)-\hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)\\ &= b(k)+\boldsymbol{m}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)\,, \end{aligned} \tag{6.80}$$

so that the MSE is:

$$\begin{aligned}J(k) &= E\{e^2(k)\}\\ &= \sigma_b^2+\boldsymbol{m}^{\mathrm{T}}(k-1)\mathbf{R}\boldsymbol{m}(k-1)\\ &= \sigma_b^2+\boldsymbol{v}^{\mathrm{T}}(k-1)\boldsymbol{\Lambda}\,\boldsymbol{v}(k-1)\\ &= \sigma_b^2+\sum_{l=1}^{L}\lambda_l(1-\mu\lambda_l)^{2k-2}v_l^2(0)\,. \end{aligned} \tag{6.81}$$

A plot of $J(k)$ versus k is called the *learning curve*. Note that the MSE decays exponentially. When the algorithm is convergent, we see that,

$$\lim_{k\to\infty} J(k) = \sigma_b^2\,. \tag{6.82}$$

This value corresponds to the MMSE, $J_{\min}$, obtained with the optimal Wiener filter when $L_{\mathrm{f}} = L$, which is what we assume in this section.

6.6.2 Stochastic Algorithm

The stochastic gradient or least-mean-square (LMS) algorithm, invented by *Widrow* and *Hoff* in the late 1950s [6.11], is certainly the most popular algorithm that we can find in the literature of adaptive filters. The popularity of the LMS is probably due to the fact that it is easy to understand, easy to implement, and robust in many respects.

One easy way to derive the stochastic gradient algorithm is by approximating the deterministic algorithm. Indeed, in practice, the two quantities $\boldsymbol{p} = E\{\boldsymbol{s}(k)x(k)\}$ and $\mathbf{R} = E\{\boldsymbol{s}(k)\boldsymbol{s}^{\mathrm{T}}(k)\}$ are in general not known. If we take their instantaneous estimates:

$$\hat{\boldsymbol{p}}(k) = \boldsymbol{s}(k)x(k)\,, \tag{6.83}$$

$$\hat{\mathbf{R}}(k) = \boldsymbol{s}(k)\boldsymbol{s}^{\mathrm{T}}(k)\,, \tag{6.84}$$

and replace them in the steepest-descent algorithm (6.58), we get:

$$\begin{aligned}\hat{\boldsymbol{h}}(k) &= \hat{\boldsymbol{h}}(k-1)+\mu[\hat{\boldsymbol{p}}(k)-\hat{\mathbf{R}}(k)\hat{\boldsymbol{h}}(k-1)]\\ &= \hat{\boldsymbol{h}}(k-1)+\mu\boldsymbol{s}(k)[x(k)-\boldsymbol{s}^{\mathrm{T}}(k)\hat{\boldsymbol{h}}(k-1)]\,. \end{aligned} \tag{6.85}$$

This simple recursion is the LMS algorithm. Contrary to the deterministic algorithm, the LMS weight vector $\hat{\boldsymbol{h}}(k)$ is now a random vector. The three following equations summarize this algorithm [6.5],

$$\hat{x}(k) = \boldsymbol{s}^{\mathrm{T}}(k)\hat{\boldsymbol{h}}(k-1)\,, \quad \text{filter output,} \tag{6.86}$$

$$e(k) = x(k)-\hat{x}(k)\,, \quad \text{error signal,} \tag{6.87}$$

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1)+\mu\boldsymbol{s}(k)e(k)\,, \quad \text{adaptation,} \tag{6.88}$$

which requires $2L$ additions and $2L+1$ multiplications at each iteration.

The stochastic gradient algorithm has been extensively studied and many theoretical results on its performance have been obtained [6.5, 10, 12]. In particular, we can show the convergence in the mean and mean square (see for example [6.13]), where under the independence assumption, the condition is remarkably the same as the one obtained for the deterministic algorithm, i. e.,

$$0 < \mu < \frac{2}{\lambda_{\max}}\,. \tag{6.89}$$

We can show that the asymptotic MSE for the LMS is:

$$\lim_{k\to\infty} J(k) = \sigma_b^2\left(1+\frac{\mu}{2}L\sigma_s^2\right)\,, \tag{6.90}$$

where $\sigma_s^2 = E\{s^2(k)\}$ is the variance of the input signal $s(k)$. If we compare (6.90) with the asymptotic MSE for the steepest-descent algorithm (6.82), we notice that a positive term,

$$J_{\mathrm{ex}}(\infty) = \frac{\mu}{2}L\sigma_s^2\sigma_b^2\,, \tag{6.91}$$

is added, called the *excess mean-square error*. This term, of course, has a negative effect on the final MSE and its appearance is due to the approximation discussed at the beginning of this subsection. We can reduce its effect by taking a very small μ, but this will increase the convergence time of the LMS. This tradeoff between fast convergence and increased MSE is a well-known effect and is something to consider in any practical implementation.

A simple condition for the stability of LMS is that,

$$|\varepsilon(k)| < |e(k)|\,, \tag{6.92}$$

Table 6.2 The normalized LMS (NLMS) algorithm

Initialization:	$\hat{\boldsymbol{h}}(0) = \boldsymbol{0}_{L\times 1}$
Parameters:	$0 < \alpha < 2$ $\delta > 0$
Error:	$e(k) = x(k) - \boldsymbol{s}^{\mathrm{T}}(k)\hat{\boldsymbol{h}}(k-1)$
Update:	$\mu(k) = \dfrac{\alpha}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)+\delta}$ $\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu(k)\boldsymbol{s}(k)e(k)$

where

$$\varepsilon(k) = x(k) - \boldsymbol{s}^{\mathrm{T}}(k)\hat{\boldsymbol{h}}(k) \tag{6.93}$$

is the a posteriori error signal, computed after the filter is updated. This makes sense intuitively since $\varepsilon(k)$ contains more meaningful information than $e(k)$.

This condition is necessary for the LMS to converge to the true impulse response but not sufficient. However, it is very useful to use here and in many other algorithms to find the bounds for the step size μ.

Inserting (6.88) into (6.93) and using the condition (6.92), we find:

$$0 < \mu < \frac{2}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} \,. \tag{6.94}$$

For L large, $\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k) = L\sigma_s^2 = \mathrm{tr}(\mathbf{R})$. On the other hand, $\mathrm{tr}(\mathbf{R}) = \sum_{l=1}^{L}\lambda_l$ and this implies that $\mathrm{tr}(\mathbf{R}) \geq \lambda_{\max}$. Hence,

$$0 < \mu < \frac{2}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} \leq \frac{2}{\lambda_{\max}} \,. \tag{6.95}$$

If we now introduce the normalized step size α ($0 < \alpha < 2$), as we did in the previous subsection, the step size of the LMS will vary with time as follows,

$$\mu(k) = \frac{\alpha}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} \,, \tag{6.96}$$

and the LMS becomes the normalized LMS (NLMS):

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \frac{\alpha\boldsymbol{s}(k)e(k)}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} \,. \tag{6.97}$$

This algorithm is extremely helpful in practice, especially with nonstationary signals, since $\mu(k)$ can adjust itself at each new iteration. In order to avoid numerical difficulties when the energy of the input signal is small, we regularize the algorithm,

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \frac{\alpha\boldsymbol{s}(k)e(k)}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)+\delta} \,, \tag{6.98}$$

where $\delta > 0$ is the regularization factor. Table 6.2 summarizes this very important algorithm. (Note that the definition of $\mu(k)$ in this table is slightly modified in order to include the regularization parameter δ.)

6.6.3 Variable-Step-Size NLMS Algorithm

The stability of the NLMS algorithm is governed by a step-size parameter. As already discussed, the choice of this parameter, within the stability conditions, reflects a tradeoff between fast convergence and good tracking ability on the one hand, and low misadjustment on the other hand. To meet these conflicting requirements, the step size needs to be controlled. While the formulation of this problem is straightforward, a good and reliable solution is not that easy to find. Many different schemes have been proposed in the last two decades [6.14–21]. In this subsection, we show how to derive in a very simple and elegant way a nonparametric variable-step-size NLMS algorithm.

We define the a priori and a posteriori error signals as, respectively,

$$\begin{aligned} e(k) &= x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\mathbf{s}(k) \\ &= \boldsymbol{s}^{\mathrm{T}}(k)[\boldsymbol{h} - \hat{\boldsymbol{h}}(k-1)] + b(k) \,, \end{aligned} \tag{6.99}$$

$$\begin{aligned} \varepsilon(k) &= x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k)\mathbf{s}(k) \\ &= \boldsymbol{s}^{\mathrm{T}}(k)[\boldsymbol{h} - \hat{\boldsymbol{h}}(k)] + b(k) \,. \end{aligned} \tag{6.100}$$

Consider the linear update equation:

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu(k)\boldsymbol{s}(k)e(k) \,. \tag{6.101}$$

One reasonable way to derive a $\mu(k)$ that makes (6.101) stable is to cancel the a posteriori error signal ([6.22] and references therein). Replacing (6.101) in (6.100) with the requirement $\varepsilon(k) = 0$ we easily find, assuming $e(k) \neq 0, \forall k$, that,

$$\mu_{\mathrm{NLMS}}(k) = \frac{1}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} \,. \tag{6.102}$$

Therefore, the obtained algorithm is the classical NLMS.

While this procedure makes sense in the absence of noise, finding the $\mu(k)$, in the presence of noise, that cancels (6.100) will introduce noise in $\hat{\boldsymbol{h}}(k)$ since

Table 6.3 The nonparametric VSS-NLMS (NPVSS-NLMS) algorithm

Initialization:	$\hat{\boldsymbol{h}}(0) = \boldsymbol{0}$
	$\hat{\sigma}_e^2(0) = 0$
Parameters:	$\lambda = 1 - \frac{1}{KL}$, exponential window with $K \geq 2$ σ_b^2, noise power known or estimated $\delta = \text{cst} \cdot \sigma_s^2$, regularization $\epsilon > 0$, very small number to avoid division by zero
Error:	$e(k) = x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)$
Update:	$\hat{\sigma}_e^2(k) = \lambda\hat{\sigma}_e^2(k-1) + (1-\lambda)e^2(k)$ $\varsigma(k) = \left[\delta + \boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)\right]^{-1}\left[1 - \frac{\sigma_b}{\epsilon + \hat{\sigma}_e(k)}\right]$ $\mu_{\text{NPVSS}}(k) = \begin{cases} \varsigma(k) & \text{if } \hat{\sigma}_e(k) \geq \sigma_b \\ 0 & \text{otherwise} \end{cases}$ $\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu_{\text{NPVSS}}(k)\boldsymbol{s}(k)e(k)$

$\boldsymbol{s}^{\mathrm{T}}(k)(\boldsymbol{h} - \hat{\boldsymbol{h}}(k)) = -b(k) \neq 0$, $\forall k$. What we would like, in fact, is to have $\boldsymbol{s}^{\mathrm{T}}(k)(\boldsymbol{h} - \hat{\boldsymbol{h}}(k)) = 0$, $\forall k$, which implies that $\varepsilon(k) = b(k)$. Hence, in this procedure we wish to find the step-size parameter $\mu(k)$ in such a way that

$$E\{\varepsilon^2(k)\} = \sigma_b^2 \,, \quad \forall k \,. \tag{6.103}$$

Using the approximation $\mathbf{s}^{\mathrm{T}}(k)\mathbf{s}(k) = L\sigma_s^2$ for $L \gg 1$, knowing that $\mu(k)$ is deterministic in nature, substituting (6.101) into (6.100), using (6.99) to eliminate $\hat{\mathbf{h}}(k-1)$, and equating to (6.103), we find:

$$\begin{aligned} E\{\varepsilon^2(k)\} &= [1 - \mu(k)L\sigma_s^2]^2\sigma_e^2(k) \\ &= \sigma_b^2 \,, \end{aligned} \tag{6.104}$$

where $\sigma_e^2(k) = E\{e^2(k)\}$ is the power of the error signal. Developing (6.104), we obtain a quadratic equation,

$$\mu^2(k) - \frac{2}{L\sigma_s^2}\mu(k) + \frac{1}{\left(L\sigma_s^2\right)^2}\left[1 - \frac{\sigma_b^2}{\sigma_e^2(k)}\right] = 0 \,, \tag{6.105}$$

for which the obvious solution is,

$$\begin{aligned} \mu_{\text{NPVSS}}(k) &= \frac{1}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)}\left[1 - \frac{\sigma_b}{\sigma_e(k)}\right] \\ &= \mu_{\text{NLMS}}(k)\alpha(k) \,, \end{aligned} \tag{6.106}$$

where $\alpha(k)$ $[0 \leq \alpha(k) \leq 1]$ is the normalized step size. Therefore, the nonparametric VSS-NLMS (NPVSS-NLMS) algorithm is [6.23],

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu_{\text{NPVSS}}(k)\boldsymbol{s}(k)e(k) \,, \tag{6.107}$$

where $\mu_{\text{NPVSS}}(k)$ is defined in (6.106).

We see from (6.106) that, before the algorithm converges, $\sigma_e(k)$ is large compared to σ_b, thus $\mu_{\text{NPVSS}}(k) \approx \mu_{\text{NLMS}}(k)$. On the other hand, when the algorithm starts to converge to the true solution, $\sigma_e(k) \approx \sigma_b$ and $\mu_{\text{NPVSS}}(k) \approx 0$. This is exactly what we desire in order to have both good convergence and low misadjustment. As we can notice, this approach was derived with almost no assumptions compared to all other algorithms belonging to the same family. Table 6.3 summarizes a practical version of the NPVSS-NLMS algorithm.

6.6.4 Proportionate NLMS Algorithms

In this subsection, we explain two very useful algorithms: the proportionate NLMS (PNLMS) and improved PNLMS (IPNLMS) algorithms.

It is well known that the NLMS algorithm converges and tracks slowly, especially for long impulse responses. In many situations where an adaptive algorithm is required, convergence and tracking are critical for a good performance of the entire system. While in the NLMS, the adaptation step is the same for all components of the filter, in the PNLMS [6.24], an adaptive individual step size is assigned to each filter coefficient. The step sizes are calculated from the last estimate of the filter coefficients in such a way that a larger coefficient receives a larger increment, thus increasing the convergence rate of that coefficient. This has the effect that active coefficients are adjusted faster than inactive coefficients (i. e., small or zero coefficients). Hence, PNLMS converges much faster than NLMS for sparse impulse responses. Unfortunately, PNLMS behaves much worse than NLMS when the impulse response is not sparse. This problem is due to the fact that the proportionate update is not very well refined. In [6.25], an IPNLMS was proposed where the adaptive individual step size has a better balance between the fixed step size of NLMS and

the large amount of proportionality in PNLMS. As a result, IPNLMS always converges and tracks better than NLMS and PNLMS, no matter how sparse the impulse response.

The error signal and the coefficient update equation of the two previously discussed algorithms can be written as:

$$e(k) = x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)\,, \tag{6.108}$$

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \frac{\alpha \mathbf{G}(k-1)\boldsymbol{s}(k)e(k)}{\boldsymbol{s}^{\mathrm{T}}(k)\mathbf{G}(k-1)\boldsymbol{s}(k) + \delta}\,, \tag{6.109}$$

where

$$\mathbf{G}(k-1) = \mathrm{diag}\{g_0(k-1)\ \ g_1(k-1) \cdots\ \ g_{L-1}(k-1)\} \tag{6.110}$$

is a diagonal matrix that adjusts the step sizes of the individual taps of the filter, α ($0 < \alpha < 2$) is the overall step-size factor, and δ is the regularization parameter.

The NLMS algorithm is obtained by taking:

$$\mathbf{G}(k) = \mathbf{I}\,, \tag{6.111}$$

$$\delta = \delta_{\mathrm{NLMS}} = \mathrm{cst}\sigma_s^2\,, \tag{6.112}$$

where $\mathbf{I}$, σ_s^2, and cst are the identity matrix, the power of the signal $s(k)$, and a small positive constant, respectively.

In the PNLMS, the diagonal elements of $\mathbf{G}(k) = \mathbf{G}_{\mathrm{p}}(k)$ are calculated as follows [6.24]:

$$\gamma_{\mathrm{p},l}(k) = \max\left\{\rho \max\left[\delta_{\mathrm{p}}, \left|\hat{h}_0(k)\right|, \cdots, \left|\hat{h}_{L-1}(k)\right|\right], \left|\hat{h}_l(k)\right|\right\}\,, \tag{6.113}$$

$$g_{\mathrm{p},l}(k) = \frac{\gamma_{\mathrm{p},l}(k)}{\|\boldsymbol{\gamma}_{\mathrm{p}}(k)\|_1}, \quad 0 \le l \le L-1\,, \tag{6.114}$$

where

$$\boldsymbol{\gamma}_{\mathrm{p}}(k) = [\gamma_{\mathrm{p},0}(k)\ \ \gamma_{\mathrm{p},1}(k)\ \cdots\ \gamma_{\mathrm{p},L-1}(k)]^{\mathrm{T}}\,.$$

The parameters δ_{p} and ρ are positive numbers with typical values $\delta_{\mathrm{p}} = 0.01$, $\rho = 0.01$. The first term in (6.113), ρ, prevents $\hat{h}_l(k)$ from stalling when its magnitude is much smaller than the magnitude of the largest coefficient and δ_{p} regularizes the updating when all coefficients are zero at initialization. For the regularization parameter, we usually choose:

$$\delta_{\mathrm{PNLMS}} = \delta_{\mathrm{NLMS}}/L\,. \tag{6.115}$$

For the IPNLMS algorithm, the diagonal matrix, $\mathbf{G}(k) = \mathbf{G}_{\mathrm{ip}}(k)$, is computed in a more-elegant way [6.25]:

$$\gamma_{\mathrm{ip},l}(k) = (1-\beta)\frac{\|\hat{\boldsymbol{h}}(k)\|_1}{L} + (1+\beta)\left|\hat{h}_l(k)\right|\,, \tag{6.116}$$

$$g_{\mathrm{ip},l}(k) = \frac{\gamma_{\mathrm{ip},l}(k)}{\|\boldsymbol{\gamma}_{\mathrm{ip}}(k)\|_1} = \frac{1-\beta}{2L} + (1+\beta)\frac{\left|\hat{h}_l(k)\right|}{2\|\hat{\boldsymbol{h}}(k)\|_1}\,, \quad 0 \le l \le L-1\,, \tag{6.117}$$

where β ($-1 \le \beta < 1$) is a parameter that controls the amount of proportionality in the IPNLMS. For $\beta = -1$, it can easily be checked that the IPNLMS and NLMS algorithms are identical. For β close to 1, IPNLMS behaves like PNLMS. In practice, a good choice for β is -0.5 or 0. With this choice and in simulations, IPNLMS always performs better than NLMS and PNLMS. As for the regularization parameter, it should be taken as:

$$\delta_{\mathrm{IPNLMS}} = \frac{1-\beta}{2L}\delta_{\mathrm{NLMS}}\,. \tag{6.118}$$

The IPNLMS algorithm is summarized in Table 6.4.

Table 6.4 The improved PNLMS (IPNLMS) algorithm

Initialization:	$\hat{h}_l(0) = 0, l = 0, 1, \cdots, L-1$
Parameters:	$-1 \le \beta < 1$ $0 < \alpha < 2$ $\delta_{\mathrm{IPNLMS}} = \mathrm{cst}\cdot\sigma_s^2\frac{1-\beta}{2L}$ $\epsilon > 0$, very small number to avoid division by zero
Error:	$e(k) = x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}(k-1)\boldsymbol{s}(k)$
Update:	$g_{\mathrm{ip},l}(k-1) = \frac{1-\beta}{2L} + (1+\beta)\frac{\left\|\hat{h}_l(k-1)\right\|}{2\left\|\hat{\boldsymbol{h}}(k-1)\right\|_1 + \epsilon}$ $\mu(k) = \frac{\alpha}{\sum_{j=0}^{L-1} s^2(k-j)g_{\mathrm{ip},j}(k-1) + \delta_{\mathrm{IPNLMS}}}$ $\hat{h}_l(k) = \hat{h}_l(k-1) + \mu(k)g_{\mathrm{ip},l}(k-1)s(k-l)e(k)$ $l = 0, 1, \cdots, L-1$

Before finishing this subsection, it is worth mentioning another variant of PNLMS, called PNLMS++ [6.26]. In this algorithm, the adaptation of the filter coefficients alternates between NLMS and PNLMS; as a result, PNLMS++ seems slightly less sensitive to the assumption of a sparse impulse response than PNLMS.

6.6.5 Sign Algorithms

Up to now, the only cost function that we have used has been the MSE. What makes this criterion so interesting is that an optimal solution (Wiener) can easily be derived as well as very powerful adaptive algorithms. An alternative to the MSE is the mean absolute error (MAE) [6.27],

$$J_{\mathrm{a}}(\hat{\boldsymbol{h}}) = E\{|e(k)|\} = E\{|x(k) - \hat{\boldsymbol{h}}^{\mathrm{T}}\boldsymbol{s}(k)|\} \,. \tag{6.119}$$

The gradient of this cost function is:

$$\frac{\partial J_{\mathrm{a}}(\hat{\boldsymbol{h}})}{\partial \hat{\boldsymbol{h}}} = -E\{\boldsymbol{s}(k)\mathrm{sgn}[e(k)]\} \,, \tag{6.120}$$

where

$$\mathrm{sgn}[e(k)] = \frac{e(k)}{|e(k)|} \,. \tag{6.121}$$

From the instantaneous value of the gradient of $J_{\mathrm{a}}(\hat{\boldsymbol{h}})$, we can derive the sign-error adaptive filter:

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu_{\mathrm{a}}\boldsymbol{s}(k)\mathrm{sgn}[e(k)] \,, \tag{6.122}$$

where μ_{a} is the adaptation step of the algorithm. This algorithm is simplified compared to the LMS since the L multiplications in the update equation are replaced by a sign change of the components of the signal vector $\boldsymbol{s}(k)$. Using the stability condition, $|\varepsilon(k)| < |e(k)|$, we deduce that:

$$0 < \mu_{\mathrm{a}} < \frac{2|e(k)|}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)} \,. \tag{6.123}$$

Another way to simplify the LMS filter is to replace $s(k)$ with its sign. We get the sign–data algorithm:

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu'_{\mathrm{a}}\mathrm{sgn}[\boldsymbol{s}(k)]e(k) \,, \tag{6.124}$$

where μ'_{a} is the adaptation step of the algorithm and the stability condition is:

$$0 < \mu'_{\mathrm{a}} < \frac{2}{\boldsymbol{s}^{\mathrm{T}}(k)\mathrm{sgn}[\boldsymbol{s}(k)]} \,. \tag{6.125}$$

Combining the two previous approaches, we derive the sign–sign algorithm:

$$\hat{\boldsymbol{h}}(k) = \hat{\boldsymbol{h}}(k-1) + \mu''_{\mathrm{a}}\mathrm{sgn}[\boldsymbol{s}(k)]\mathrm{sgn}[e(k)] \,, \tag{6.126}$$

for which the stability condition is:

$$0 < \mu''_{\mathrm{a}} < \frac{2|e(k)|}{\boldsymbol{s}^{\mathrm{T}}(k)\mathrm{sgn}[\boldsymbol{s}(k)]} \,. \tag{6.127}$$

The algorithms derived in this subsection are very simple to implement and can be very useful in some applications. However, their convergence rate is usually slower than the LMS and their excess MSE is higher [6.28–30].

6.7 MIMO Wiener Filter

In this section, we consider a MIMO system with M inputs and N outputs (see Sect. 6.2 for more details):

$$\boldsymbol{x}(k) = \mathbf{H}\boldsymbol{s}(k) + \boldsymbol{b}(k) \,, \tag{6.128}$$

where

$$x_n(k) = \sum_{m=1}^{M} \boldsymbol{h}_{nm}^{\mathrm{T}}\boldsymbol{s}_m(k) + b_n(k) = \boldsymbol{h}_{n:}^{\mathrm{T}}\boldsymbol{s}(k) + b_n(k), \quad n = 1, 2, \cdots, N \,, \tag{6.129}$$

and

$$\mathbf{H} = \begin{pmatrix} \boldsymbol{h}_{11}^{\mathrm{T}} & \boldsymbol{h}_{12}^{\mathrm{T}} & \cdots & \boldsymbol{h}_{1M}^{\mathrm{T}} \\ \boldsymbol{h}_{21}^{\mathrm{T}} & \boldsymbol{h}_{22}^{\mathrm{T}} & \cdots & \boldsymbol{h}_{2M}^{\mathrm{T}} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{h}_{N1}^{\mathrm{T}} & \boldsymbol{h}_{N2}^{\mathrm{T}} & \cdots & \boldsymbol{h}_{NM}^{\mathrm{T}} \end{pmatrix}_{N\times ML} = \begin{pmatrix} \boldsymbol{h}_{1:}^{\mathrm{T}} \\ \boldsymbol{h}_{2:}^{\mathrm{T}} \\ \vdots \\ \boldsymbol{h}_{N:}^{\mathrm{T}} \end{pmatrix} \,. \tag{6.130}$$

We define the error signal at time k at the n-th output as:

$$e_n(k) = x_n(k) - \hat{x}_n(k) = x_n(k) - \hat{\boldsymbol{h}}_{n:}^{\mathrm{T}}\boldsymbol{s}(k) \,, \quad n = 1, 2, \cdots, N \,, \tag{6.131}$$

Table 6.5 The MISO NLMS algorithm

Initialization:	$\hat{\boldsymbol{h}}_{n:}(0) = \mathbf{0}_{ML\times 1}$
Parameters:	$0 < \alpha < 2$
	$\delta > 0$
Error:	$e_n(k) = x_n(k) - \boldsymbol{s}^{\mathrm{T}}(k)\hat{\boldsymbol{h}}_{n:}(k-1)$
Update:	$\mu(k) = \dfrac{\alpha}{\boldsymbol{s}^{\mathrm{T}}(k)\boldsymbol{s}(k)+\delta}$
	$\hat{\boldsymbol{h}}_{n:}(k) = \hat{\boldsymbol{h}}_{n:}(k-1) + \mu(k)\boldsymbol{s}(k)e_n(k)$

where $\hat{\boldsymbol{h}}_{n:}$ is an estimate of $\boldsymbol{h}_{n:}$. It is more convenient to define an error signal vector for all outputs:

$$\begin{aligned}\boldsymbol{e}(k) &= \boldsymbol{x}(k) - \hat{\boldsymbol{x}}(k)\\ &= \boldsymbol{x}(k) - \hat{\mathbf{H}}\boldsymbol{s}(k)\,,\end{aligned} \tag{6.132}$$

where $\hat{\boldsymbol{H}}$ is an estimate of $\boldsymbol{H}$ and

$$\boldsymbol{e}(k) = [e_1(k)\;\; e_2(k)\;\; \cdots\;\; e_N(k)]^{\mathrm{T}}\,.$$

Having written the error signal, we now define the MIMO MSE with respect to the modeling filters as:

$$\begin{aligned}J(\hat{\mathbf{H}}) &= E\{\boldsymbol{e}^{\mathrm{T}}(k)\boldsymbol{e}(k)\}\\ &= \sum_{n=1}^{N} E\left\{e_n^2(k)\right\} = \sum_{n=1}^{N} J_n(\hat{\boldsymbol{h}}_{n:})\,.\end{aligned} \tag{6.133}$$

The minimization of (6.133) leads to the MIMO Wiener–Hopf equations:

$$\mathbf{R}_{ss}\hat{\mathbf{H}}_{\mathrm{o}}^{\mathrm{T}} = \mathbf{P}_{sx}\,, \tag{6.134}$$

where

$$\begin{aligned}\mathbf{R}_{ss} &= E\{\boldsymbol{s}(k)\boldsymbol{s}^{\mathrm{T}}(k)\}\\ &= \begin{pmatrix}\mathbf{R}_{s_1s_1} & \mathbf{R}_{s_1s_2} & \cdots & \mathbf{R}_{s_1s_M}\\ \mathbf{R}_{s_2s_1} & \mathbf{R}_{s_2s_2} & \cdots & \mathbf{R}_{s_2s_M}\\ \vdots & \vdots & \ddots & \vdots\\ \mathbf{R}_{s_Ms_1} & \mathbf{R}_{s_Ms_2} & \cdots & \mathbf{R}_{s_Ms_M}\end{pmatrix}\end{aligned} \tag{6.135}$$

is the input signal covariance matrix (which has a block-Toeplitz structure) with $\mathbf{R}_{s_ms_i} = E\{\boldsymbol{s}_m(k)\boldsymbol{s}_i^{\mathrm{T}}(k)\}$, and

$$\begin{aligned}\mathbf{P}_{sx} &= E\{\boldsymbol{s}(k)\boldsymbol{x}^{\mathrm{T}}(k)\}\\ &= (\boldsymbol{p}_{sx_1}\;\; \boldsymbol{p}_{sx_2}\;\; \cdots\;\; \boldsymbol{p}_{sx_N})\end{aligned} \tag{6.136}$$

is the cross-correlation matrix between the input and output signals, with $\mathbf{p}_{sx_n} = E\{\boldsymbol{s}(k)x_n(k)\}$.

It can easily be seen that the MIMO Wiener–Hopf equations (6.134) can be decomposed into N independent MISO Wiener–Hopf equations,

$$\mathbf{R}_{ss}\hat{\boldsymbol{h}}_{n:,\mathrm{o}} = \boldsymbol{p}_{sx_n}\,,\quad n = 1, 2, \cdots, N\,, \tag{6.137}$$

each one corresponding to an output signal of the system. In other words, minimizing $J(\hat{\mathbf{H}})$ or minimizing each $J_n(\hat{\boldsymbol{h}}_{n:})$ independently gives exactly the same results from an identification point of view. This observation is very important from a practical point of view when adaptive algorithms need to be designed. Indeed, any MIMO adaptive filter is simplified to N MISO adaptive filters. As an example, we give the MISO NLMS algorithm in Table 6.5. We deduce from this discussion that, obviously, the identification of a SIMO system is equivalent to the identification of N independent SISO systems. As a result, with a reference signal, the identification of any acoustic system simplifies to the identification of SISO or MISO systems.

6.7.1 Conditioning of the Covariance Matrix

The best possible case for the identification of a MISO system is when the input signals $s_m(k)$, $m = 1, 2, \cdots, M$, are uncorrelated. In this scenario, we have:

$$\mathbf{R}_{s_ms_i} = \mathbf{0}_{L\times L}\,,\quad \forall m, i = 1, 2, \cdots, M,\; m \neq i\,, \tag{6.138}$$

and the input signal covariance matrix $\mathbf{R}_{ss}$ is block-diagonal. Therefore, if $\mathbf{R}_{s_ms_m}$, $m = 1, 2, \cdots, M$, are nonsingular and well conditioned, the impulse responses of the MISO system are easy to estimate. This case, however, does not often occur in practice so it is of little interest.

The worst possible case, from an identification point of view, is when the signals $s_m(k)$ are generated from a unique source $s_{\mathrm{s}}(k)$, i. e.,

$$s_m(k) = \boldsymbol{g}_m^{\mathrm{T}}s_{\mathrm{s}}(k)\,,\quad m = 1, 2, \cdots, M\,, \tag{6.139}$$

where

$$\boldsymbol{g}_m = [g_{m,0}\;\; g_{m,1}\;\; \cdots\;\; g_{m,L-1}]^{\mathrm{T}}$$

is the impulse response between the source $s_{\mathrm{s}}(k)$ and the signal $s_m(k)$. In this scenario, it can be shown that matrix $\mathbf{R}_{ss}$ is rank-deficient by, at least, $(M-2)L+1$.

As a matter of fact, from (6.139), the input signal vector $\boldsymbol{s}_\mathrm{s}(k)$ can be written as

$$
\begin{aligned}
\mathbf{s}(k) &= \left[\boldsymbol{s}_1^\mathrm{T}(k)\ \boldsymbol{s}_2^\mathrm{T}(k)\ \cdots\ \boldsymbol{s}_M^\mathrm{T}(k)\right]^\mathrm{T} \\
&= \mathbf{G}\boldsymbol{s}_\mathrm{s}(k) \\
&= \begin{pmatrix}
g_{1,0} & g_{1,1} & \cdots & g_{1,L-1} & 0 & 0 & \cdots & 0 \\
0 & g_{1,0} & g_{1,1} & \cdots & g_{1,L-1} & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & g_{1,0} & g_{1,1} & \cdots & & g_{1,L-1} \\
g_{2,0} & g_{2,1} & \cdots & g_{2,L-1} & 0 & 0 & \cdots & 0 \\
0 & g_{2,0} & g_{2,1} & \cdots & g_{2,L-1} & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & g_{2,0} & g_{2,1} & \cdots & & g_{2,L-1} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots \\
g_{M,0} & g_{M,1} & \cdots & g_{M,L-1} & 0 & 0 & \cdots & 0 \\
0 & g_{M,0} & g_{M,1} & \cdots & g_{M,L-1} & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & g_{M,0} & g_{M,1} & \cdots & & g_{M,L-1}
\end{pmatrix} \\
&\quad \cdot \begin{pmatrix}
s_\mathrm{s}(k) \\
s_\mathrm{s}(k-1) \\
s_\mathrm{s}(k-2) \\
\vdots \\
s_\mathrm{s}(k-2L+3) \\
s_\mathrm{s}(k-2L+2)
\end{pmatrix},
\end{aligned}
$$

where $\mathbf{G}$ is an $ML\times(2L-1)$ matrix, containing the impulse responses g_m and is assumed to be of full column rank, and $\boldsymbol{s}_\mathrm{s}(k)$ is a $(2L-1)\times 1$ vector. We then have:

$$
\begin{aligned}
\mathbf{R}_{ss} &= E\{\mathbf{s}(k)\mathbf{s}^\mathrm{T}(k)\} \\
&= \mathbf{G}E\left\{\boldsymbol{s}_\mathrm{s}(k)\boldsymbol{s}_\mathrm{s}^\mathrm{T}(k)\right\}\mathbf{G}^\mathrm{T} \\
&= \mathbf{G}\mathbf{R}_{s_\mathrm{s}s_\mathrm{s}}\mathbf{G}^\mathrm{T}\,, \qquad (6.140)
\end{aligned}
$$

where $\mathbf{R}_{s_\mathrm{s}s_\mathrm{s}}$ is the source signal covariance matrix of size $(2L-1)\times(2L-1)$, assumed to be full rank. We immediately see from (6.140), that:

$$
\begin{aligned}
\mathrm{Rank}[\mathbf{R}_{ss}] &= \min\{\mathrm{Rank}[\mathbf{G}],\ \mathrm{Rank}[\mathbf{R}_{s_\mathrm{s}s_\mathrm{s}}]\} \\
&= 2L-1\,, \qquad (6.141)
\end{aligned}
$$

and

$$
\begin{aligned}
\mathrm{Null}[\mathbf{R}_{ss}] &= ML - \mathrm{Rank}[\mathbf{R}_{ss}] \\
&= (M-2)L+1\,, \qquad (6.142)
\end{aligned}
$$

where Null[] and Rank[] denote the dimension of the null space and the rank of a matrix, respectively.

From this analysis, one can see that a MISO system is rank deficient if its inputs are the filtered version of the same source signal. Thus, the Wiener–Hopf equations do not have a unique solution.

In most practical situations, the signals $s_m(k)$, $m=1,2,\cdots,M$, are somehow related. If they are highly coherent, adaptive algorithms will be very slow to converge to the true solution and in some situations, they will converge to a solution that is far from the desired one.

We are now going to show in the particular case of a MISO system with two inputs how a high coherence between these signals affects the condition number of the covariance matrix:

$$
\mathbf{R}_{ss} = \begin{pmatrix}
\mathbf{R}_{s_1s_1} & \mathbf{R}_{s_1s_2} \\
\mathbf{R}_{s_2s_1} & \mathbf{R}_{s_2s_2}
\end{pmatrix}.
$$

For $L\to\infty$, a Toeplitz matrix is asymptotically equivalent to a circulant matrix if its elements are absolutely summable [6.31], which is the case for speech signals. In this situation, we can decompose

$$
\mathbf{R}_{s_ms_i} = \mathbf{F}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{s_ms_i}\mathbf{F}\,, \quad m,i=1,2\,, \qquad (6.143)
$$

where $\mathbf{F}$ is the Fourier matrix and the diagonal matrix

$$
\begin{aligned}
\underset{\Rightarrow}{\mathbf{R}}_{s_ms_i} &= \mathrm{diag}\left\{\underset{\Rightarrow}{\mathbf{R}}_{s_ms_i}(0),\ \underset{\Rightarrow}{\mathbf{R}}_{s_ms_i}(1),\right. \\
&\qquad \left.\cdots,\ \underset{\Rightarrow}{\mathbf{R}}_{s_ms_i}(L-1)\right\} \qquad (6.144)
\end{aligned}
$$

contains elements corresponding to the L frequency bins that are formed from the discrete Fourier transform (DFT) of the first column of $\mathbf{R}_{s_ms_i}$. Letting $r_{s_ms_i}(l)$ be the auto- and cross-correlation for $m=i$ and $m\neq i$, respectively, we see that the spectral content between two signals is related to the correlation function by

$$
\begin{aligned}
\underset{\Rightarrow}{\mathbf{R}}_{s_ms_i}(f) &= \sum_{l=-\infty}^{\infty} r_{s_ms_i}(l)e^{-\mathrm{i}2\pi fl}\,, \\
&\quad f=0,1,\cdots,L-1\,. \qquad (6.145)
\end{aligned}
$$

Using (6.143), $\mathbf{R}_{ss}$ can be expressed in terms of its spectra as:

$$\begin{aligned}\mathbf{R}_{ss} &= \mathbf{F}_{\mathrm{d}}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{ss}\mathbf{F}_{\mathrm{d}} \\ &= \begin{pmatrix}\mathbf{F}^{-1} & \mathbf{0}_{L\times L}\\ \mathbf{0}_{L\times L} & \mathbf{F}^{-1}\end{pmatrix}\begin{pmatrix}\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1} & \underset{\Rightarrow}{\mathbf{R}}_{s_1s_2}\\ \underset{\Rightarrow}{\mathbf{R}}_{s_2s_1} & \underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}\end{pmatrix} \\ &\quad\cdot\begin{pmatrix}\mathbf{F} & \mathbf{0}_{L\times L}\\ \mathbf{0}_{L\times L} & \mathbf{F}\end{pmatrix}. \end{aligned} \tag{6.146}$$

To compute the condition number $\chi_{\mathrm{F}}^2(\mathbf{R}_{ss}^{1/2})$ (Sect. 6.5), we need to compute $\mathrm{tr}(\mathbf{R}_{ss})$ and $\mathrm{tr}(\mathbf{R}_{ss}^{-1})$. The first trace is easy to compute. Indeed, using (6.146), we easily find:

$$\begin{aligned}\mathrm{tr}(\mathbf{R}_{ss}) &= \mathrm{tr}\left(\mathbf{F}_{\mathrm{d}}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{ss}\mathbf{F}_{\mathrm{d}}\right) = \mathrm{tr}\left(\underset{\Rightarrow}{\mathbf{R}}_{ss}\right) \\ &= \sum_{l=0}^{L-1}\left(\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}(l)+\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}(l)\right). \end{aligned} \tag{6.147}$$

For the second trace, we have:

$$\mathrm{tr}\left(\mathbf{R}_{ss}^{-1}\right) = \mathrm{tr}\left(\mathbf{F}_{\mathrm{d}}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{ss}^{-1}\mathbf{F}_{\mathrm{d}}\right) = \mathrm{tr}\left(\underset{\Rightarrow}{\mathbf{R}}_{ss}^{-1}\right). \tag{6.148}$$

Furthermore, it is easy to show that:

$$\begin{aligned}\underset{\Rightarrow}{\mathbf{R}}_{ss}^{-1} &= \begin{pmatrix}\underset{\Rightarrow}{\mathbf{R}}_1^{-1} & \mathbf{0}_{L\times L}\\ \mathbf{0}_{L\times L} & \underset{\Rightarrow}{\mathbf{R}}_2^{-1}\end{pmatrix} \\ &\quad\cdot\begin{pmatrix}\mathbf{I}_{L\times L} & -\underset{\Rightarrow}{\mathbf{R}}_{s_1s_2}\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}^{-1}\\ -\underset{\Rightarrow}{\mathbf{R}}_{s_2s_1}\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}^{-1} & \mathbf{I}_{L\times L}\end{pmatrix}, \end{aligned} \tag{6.149}$$

where

$$\underset{\Rightarrow}{\mathbf{R}}_1 = \left[\mathbf{I}_{L\times L} - \underset{\Rightarrow}{\mathbf{R}}_{s_1s_2}^2\left(\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}^{-1}\right)\right]\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}, \tag{6.150}$$

$$\underset{\Rightarrow}{\mathbf{R}}_2 = \left[\mathbf{I}_{L\times L} - \underset{\Rightarrow}{\mathbf{R}}_{s_1s_2}^2\left(\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}^{-1}\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}^{-1}\right)\right]\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}. \tag{6.151}$$

Hence,

$$\begin{aligned}\mathrm{tr}\left(\mathbf{R}_{ss}^{-1}\right) &= \mathrm{tr}\left(\underset{\Rightarrow}{\mathbf{R}}_1^{-1}+\underset{\Rightarrow}{\mathbf{R}}_2^{-1}\right) \\ &= \sum_{l=0}^{L-1}(1-|\gamma(l)|^2)^{-1} \\ &\quad\times\left[\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}^{-1}(l)+\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}^{-1}(l)\right], \end{aligned} \tag{6.152}$$

where

$$\begin{aligned}|\gamma(f)|^2 &= \frac{\left|\underset{\Rightarrow}{\mathbf{R}}_{s_1s_2}(f)\right|^2}{\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}(f)\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}(f)}, \\ f &= 0, 1, \cdots, L-1, \end{aligned} \tag{6.153}$$

is the squared interchannel coherence function of the f-th frequency bin.

We finally obtain the relationship between the interchannel coherence and the condition number based on the Frobenius norm:

$$\begin{aligned}\chi_{\mathrm{F}}^2\left(\mathbf{R}_{ss}^{1/2}\right) &= \left\{\sum_{l=0}^{L-1}\left[\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}(l)+\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}(l)\right]\right\} \\ &\quad\times\left\{\sum_{l=0}^{L-1}[1-|\gamma(l)|^2]^{-1}\left[\underset{\Rightarrow}{\mathbf{R}}_{s_1s_1}^{-1}(l)\right.\right. \\ &\quad\left.\left.+\,\underset{\Rightarrow}{\mathbf{R}}_{s_2s_2}^{-1}(l)\right]\right\}. \end{aligned} \tag{6.154}$$

It is now evident from the previous expression that $\chi_{\mathrm{F}}^2(\mathbf{R}_{ss}^{1/2})$ increases with the squared interchannel coherence function, hence degrading the condition of $\mathbf{R}_{ss}$; as $\gamma \to 1$, $\chi_{\mathrm{F}}^2(\mathbf{R}_{ss}^{1/2}) \to \infty$ and the identification of the system is increasingly difficult, if not impossible.

6.8 Conclusions

In this chapter, we have explained the most important results of the Wiener theory in the context of system identification.

After discussing the four basic signal models, we derived the optimal Wiener filter for a SISO system and showed that this filter can be a very good approximation of the desired impulse response.

We discussed in details the condition number of the input signal correlation matrix. This matrix appears explicitly in the Wiener–Hopf equations and implicitly in all adaptive filters. A high condition number will perturb the accuracy of the solution of the Wiener–Hopf equations and will slow the rate of convergence of most adaptive algorithms. A fast, efficient algorithm to compute the conditional number was also developed.

We also discussed several important adaptive filters. In particular, the NLMS algorithm, which is extremely popular and useful in practice, was derived. Other emerging algorithms, such as the IPNLMS, were presented.

We generalized the Wiener principle to the MIMO system case. We showed that the MIMO Wiener–Hopf

equations can be decomposed into N independent MISO Wiener–Hopf equations. As a result, adaptive filters for SISO and MISO systems, with a reference signal, cover all possible cases. A deep analysis of the conditioning of the input signal covariance matrix was given, showing that identification is not always obvious and depends on the interchannel coherence between the input signals.

References

6.1 N. Wiener: *Extrapolation, Interpolation, and Smoothing of Stationary Time Series* (Wiley, New York 1949)

6.2 N. Wiener, E. Hopf: On a class of singular integral equations, Proc. Prussian Acad. Math.-Phys. Ser. (1931) p. 696

6.3 N. Levinson: The Wiener rms (root-mean-square) error criterion in filter design and prediction, J. Math. Phys. **25**, 261–278 (1947)

6.4 T. Kailath: A view of three decades of linear filtering theory, IEEE Trans. Inf. Theory **IT-20**, 146–181 (1974)

6.5 S. Haykin: *Adaptive Filter Theory*, 4th edn. (Prentice Hall, Upper Saddle River 2002)

6.6 J. Benesty, T. Gänsler: Computation of the condition number of a non-singular symmetric Toeplitz matrix with the Levinson–Durbin algorithm, IEEE Trans. Signal Process. **54**, 2362–2364 (2006)

6.7 J. Benesty, T. Gänsler: New insights into the RLS algorithm, EURASIP J. Appl. Signal Process. **2004**, 331–339 (2004)

6.8 G.H. Golub, C.F. Van Loan: *Matrix Computations* (The Johns Hopkins Univ. Press, Baltimore 1996)

6.9 J.M.B. Dias, J.M.N. Leitão: Efficient computation of $\mathrm{tr}\mathbf{TR}^{-1}$ for Toeplitz matrices, IEEE Signal Process. Let. **9**, 54–56 (2002)

6.10 B. Widrow: Adaptive filters. In: *Aspects of Network and System Theory*, ed. by R.E. Kalman, N. DeClaris (Holt Rinehart and Winston, New York 1970)

6.11 B. Widrow, M.E. Hoff Jr.: Adaptive switching circuits, IRE WESCON Conv. Rec. **4**, 96–104 (1960)

6.12 A. Feuer, E. Weinstein: Convergence analysis of LMS filters with uncorrelated Gaussian data, IEEE Trans. Acoust. Speech ASSP **33**, 222–230 (1985)

6.13 B. Widrow, S.D. Stearns: *Adaptive Signal Processing* (Prentice Hall, Englewood Cliffs 1985)

6.14 R.W. Harris, D.M. Chabries, F.A. Bishop: A variable step (VS) adaptive filter algorithm, IEEE Trans. Acoust. Speech ASSP **34**, 309–316 (1986)

6.15 R.H. Kwong, E.W. Johnston: A variable step size LMS algorithm, IEEE Trans. Signal Process. **40**, 1633–1642 (1992)

6.16 V.J. Mathews, Z. Xie: A stochastic gradient adaptive filter with gradient adaptive step size, IEEE Trans. Signal Process. **41**, 2075–2087 (1993)

6.17 J.B. Evans, P. Xue, B. Liu: Analysis and implementation of variable step size adaptive algorithms, IEEE Trans. Signal Process. **41**, 2517–2535 (1993)

6.18 T. Aboulnasr, K. Mayyas: A robust variable step-size LMS-type algorithm: analysis and simulations, IEEE Trans. Signal Process. **45**, 631–639 (1997)

6.19 D.I. Pazaitis, A.G. Constantinides: A novel kurtosis driven variable step-size adaptive algorithm, IEEE Trans. Signal Process. **47**, 864–872 (1999)

6.20 A. Mader, H. Puder, G.U. Schmidt: Step-size control for acoustic echo cancellation filters – An overview, Signal Process. **80**, 1697–1719 (2000)

6.21 H.-C. Shin, A.H. Sayed, W.-J. Song: Variable step-size NLMS and affine projection algorithms, IEEE Signal Process. Lett. **11**, 132–135 (2004)

6.22 D.R. Morgan, S.G. Kratzer: On a class of computationally efficient, rapidly converging, generalized NLMS algorithms, IEEE Signal Process. Lett. **3**, 245–247 (1996)

6.23 J. Benesty, H. Rey, L.R. Vega, S. Tressens: A nonparametric VSS-NLMS algorithm, IEEE Signal Process. Lett. **13**, 581–584 (2006), .

6.24 D.L. Duttweiler: Proportionate normalized least-mean-square adaptation in echo cancelers, IEEE Trans. Audio Speech **8**, 508–518 (2000)

6.25 J. Benesty, S.L. Gay: An improved PNLMS algorithm, Proc. IEEE ICASSP (2002) pp. 1881–1884

6.26 S.L. Gay: An efficient fast converging adaptive filter for network echo cancellation, Proc. Assilomar Conf. **1**, 394–398 (1998)

6.27 A. Gersho: Adaptive filtering with binary reinforcement, IEEE Trans. Inf. Theory **IT-30**, 191–199 (1984)

6.28 M.G. Bellanger: *Adaptive Digital Filters and Signal Analysis* (Marcel Dekker, New York 1987)

6.29 T. Claasen, W. Mecklenbrauker: Comparison of the convergence of two algorithms for adaptive FIR digital filters, IEEE Trans. Acoust. Speech ASSP **29**, 670–678 (1981)

6.30 N.J. Bershad: On the optimum data non-linearity in LMS adaptation, IEEE Trans. Acoust. Speech ASSP **34**, 69–76 (1986)

6.31 R. Gray: On the asymptotic eigenvalue distribution of Toeplitz matrices, IEEE Trans. Inform. Theory **IT-18**, 725–730 (1972)

17. Analysis-by-Synthesis Speech Coding

J.-H. Chen, J. Thyssen

Since the early 1980s, advances in speech coding technologies have enabled speech coders to achieve bit-rate reductions of a factor of 4 to 8 while maintaining roughly the same high speech quality. One of the most important driving forces behind this feat is the so-called *analysis-by-synthesis* paradigm for coding the excitation signal of predictive speech coders. In this chapter, we give an overview of many variations of the analysis-by-synthesis excitation coding paradigm as exemplified by various speech coding standards around the world. We describe the variations of the same basic theme in the context of different coder structures where these techniques are employed. We also attempt to show the relationship between them in the form of a *family tree*. The goal of this chapter is to give the readers a big-picture understanding of the dominant types of analysis-by-synthesis excitation coding techniques for predictive speech coding.

17.1 Overview

Historically, speech coding technology has been dominated by coders based on linear prediction. To achieve good speech quality, most speech coding standards are *waveform-approximating coders* (or *waveform coders* for short), and the majority use linear prediction to exploit the redundancy in the speech waveform.

Of course, the techniques in speech coding standards are not the only ones available, as there are many other speech coding techniques proposed in the literature that have not been used in standards. However, speech coding standards represent the dominant and most widely deployed speech coding techniques. Often they also include the best-performing techniques for the given requirements of bit rate, coding delay, and codec complexity at the time they were standardized. Therefore, it is useful to examine some of the techniques in speech coding standards to see how the dominant speech coding techniques have evolved over the years. For a comprehensive review of speech coding standards up to about 1995, see [17.1].

The first speech coding standard based on predictive waveform coding is probably 32 kb/s adaptive differential pulse code modulation (ADPCM), which was initially standardized by the Comité Consultatif International Téléphonique et Télégraphique (CCITT, the predecessor of the ITU-T) as recommendation G.721 in 1984 and later modified and restandardized as part of G.726 in 1986. The G.726 standard is a narrow-band (telephone-bandwidth) speech coder with an input sampling rate of 8 kHz and encoding bit rates of 16, 24, 32, and 40 kb/s, with 32 kb/s being the most widely used bit rate of G.726.

The basic idea of the G.726 ADPCM coder is to use an adaptive linear predictor to predict the input speech signal, and then use an adaptive scalar quantizer to quantize the difference signal between the input speech and the predicted version. Since statistically this difference signal tends to be smaller than the input speech signal, one can use a lower bit rate to quantize the difference signal to the same precision as direct quantization of the speech signal itself. This is how predictive coders achieve bit-rate reduction through the use of prediction to exploit the redundancy between nearby speech samples. The difference signal is often called the *prediction residual signal*, because it is the residual of the speech signal after the predictable portion is removed. In analysis-by-synthesis speech coders, this prediction residual signal is often referred to as the *excitation signal*. The reason will become clear during later discussions in this chapter.

In 1982, at about the same time that G.721 ADPCM was being developed, *Atal* and *Remde* proposed the first analysis-by-synthesis speech waveform coding technique called *multipulse linear predictive coding* (MPLPC) [17.2]. This work led to *Atal* and *Schroeder*'s 1984 proposal of another analysis-by-synthesis speech coding technique called *stochastic coding* [17.3], which was renamed *code-excited linear prediction* (CELP) [17.4] in 1985. From then on, CELP became the dominant speech coding technique for the next two decades, with hundreds or perhaps even thousands of technical papers published on variations of CELP techniques.

Due to their high coding efficiency, variations of CELP coding techniques together with other advancements have enabled speech waveform coders to halve the bit rate of 32 kb/s G.726 ADPCM three times while maintaining roughly the same speech quality. This is evidenced by the ITU-T's speech coding standardization effort after standardizing 32 kb/s ADPCM in 1984.

The first halving of bit rate to 16 kb/s happened with the ITU-T G.728 low-delay CELP (LD-CELP) coder [17.5, 6]. At the cost of increasing the buffering delay from one to five samples (0.625 ms at 8 kHz sampling rate), the G.728 coder halved the bit rate of 32 kb/s G.726 ADPCM while maintaining equivalent or better speech quality for all conditions tested.

The second halving of bit rate to 8 kb/s happened with the ITU-T G.729 conjugate-structure algebraic CELP (CS-ACELP) [17.7]. At the cost of increasing the buffering delay further to 80 samples of frame size plus 40 samples of look-ahead (or 15 ms total), the G.729 coder halved the bit rate again to 8 kb/s while maintaining the speech quality for most test conditions except tandeming and speech in background noise.

The third halving of bit rate to 4 kb/s happened with candidate coders [17.8, 9] submitted for the ITU-T's 4 kb/s speech coding standardization. A promising candidate 4 kb/s coder [17.8] was based on CELP. Although the ITU-T eventually did not standardize a 4 kb/s coder, this CELP-based candidate coder did achieve equivalent speech quality as 32 kb/s G.726 ADPCM at least for clean speech conditions at the cost of further increasing the buffering delay to 30–35 ms.

Of course, the G.728, G.729, and ITU-T 4 kb/s candidate coders are merely three example coders that help to make a point that the analysis-by-synthesis

speech coding technique was the main driving force behind the feat of reducing the bit rate by a factor of 4 to 8 within two decades while maintaining equivalent speech quality, at least for clean speech. In the last two decades, researchers have proposed numerous other speech waveform coders in the bit-rate range of 4–16 kb/s that achieve speech quality equivalent to or better than the speech quality of the 32 kb/s G.726 ADPCM coder. The vast majority of these are analysis-by-synthesis speech coders based on either CELP or MPLPC. Therefore, it is fair to say that *analysis-by-synthesis* is undeniably the most important speech coding method in modern low-bit-rate speech waveform coding.

The main emphasis of this chapter is to describe many of the major flavors of analysis-by-synthesis excitation coding for linear predictive speech waveform coders, as exemplified by various speech coding standards around the world. Due to the space limitation, it is not possible to cover all speech coding standards or even all aspects of the standards discussed in this chapter. Readers interested in learning more about other aspects of analysis-by-synthesis speech coders are referred to [17.10].

The rest of this chapter is organized as follows. Section 17.2 describes the basic concepts of the analysis-by-synthesis paradigm in the context of speech coding. Section 17.3 gives an overview of the different flavors of analysis-by-synthesis excitation coding and shows the relationship between them in the form of a *family tree*. Sections 17.4–17.17 then provide somewhat more-detailed descriptions of these different flavors of analysis-by-synthesis speech coders one-by-one, including MPLPC, original CELP, regular-pulse excitation (RPE), federal standard 1016 (FS1016) CELP, vector sum excited linear prediction (VSELP), LD-CELP, pitch synchronous innovation CELP (PSI-CELP), ACELP, CS-ACELP, relaxed CELP (RCELP), extended CELP (eX-CELP), forward backward linear predictive coding (FB-LPC), two-stage noise feedback coding (TSNFC), and embedded CELP, roughly in a chronological order. Section 17.18 summarizes these analysis-by-synthesis speech coders in a table format. Finally, Sect. 17.19 concludes this chapter.

17.2 Basic Concepts of Analysis-by-Synthesis Coding

17.2.1 Definition of Analysis-by-Synthesis

What exactly is *analysis-by-synthesis*? The phrase "*analysis-by-synthesis*" can be traced back to at least 1959 in a paper by *Halle* and *Stevens* [17.11]. In another 1961 paper by *Bell* et al. [17.12], a detailed definition of analysis-by-synthesis is given as follows:

> *The term analysis-by-synthesis is used to refer to an active analysis process that can be applied to signals that are produced by a generator whose properties are known. The heart of an analysis-by-synthesis system is a signal generator capable of synthesizing all and only the signals to be analyzed. The signals synthesized by the generator are compared with the signals to be analyzed, and a measure of error is computed. Different signals are generated until one is found that causes the error to reach some smallest value, at which time the analyzer indicates the properties of the internally generated signal.*

Although this definition was given four and a half decades ago in a paper discussing the subject of speech analysis, it is still equally valid today as a description of the basic concepts behind analysis-by-synthesis speech waveform coding.

17.2.2 From Conventional Predictive Waveform Coding to a Speech Synthesis Model

To understand analysis-by-synthesis speech waveform coding, it is useful to look first at conventional predictive speech waveform coding. In conventional predictive speech coders based on linear prediction, the prediction is typically performed using a weighted sum of previously quantized speech samples rather than unquantized input speech samples. This is because the quantized speech samples are available at the decoder while the input speech samples are not. Performing linear prediction based on the quantized speech in the encoder enables the decoder to produce the same predicted speech and track the encoder states in the absence of transmission errors. Prediction based on previously quantized speech signal is called *closed-loop* prediction, while prediction based on unquantized input speech signal is called *open-loop* prediction. For in-depth discussion of predictive waveform coding principles, see [17.13].

The encoder and decoder of the most basic form of closed-loop predictive waveform coding is shown in Fig. 17.1. In the encoder in Fig. 17.1a, the linear predictor, represented by the transfer function $P(z)$,

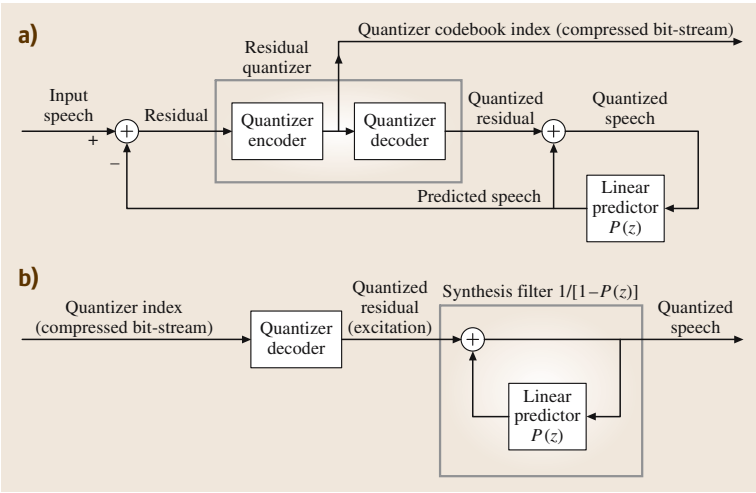


Fig. 17.1a,b Basic structure of conventional linear predictive speech waveform coder. **(a)** Speech encoder, and **(b)** speech decoder

uses previously quantized speech signal as its input to produce the predicted speech signal, which is subtracted from the input speech signal to get the prediction residual signal. After the prediction residual signal is quantized by the quantizer, the same predicted speech signal is added back to the quantized prediction residual signal to get the quantized speech signal. The sequence of quantizer codebook indices corresponding to the sequence of selected quantizer output levels is transmitted to the decoder as the compressed bitstream.

Note that the decoder structure in Fig. 17.1b is basically just a replica of the right half of the encoder structure. Therefore, if the compressed bitstream is received without transmission errors, then the decoder can decode the same quantized speech signal as the quantized speech signal in the encoder. Note that the feedback loop in the decoder that contains the linear predictor can be regarded as a *synthesis filter* with a transfer function of $1/[1-P(z)]$. This synthesis filter corresponds to an autoregressive model. It should also be noted that the quantized prediction residual at the output of the quantizer decoder can be regarded as the *excitation source* for this synthesis filter. From this perspective, the decoder of a conventional linear predictive speech coder can be viewed as a *source-filter model* [17.14] for speech production or speech synthesis. Therefore, the task of the corresponding encoder is to determine the quantized excitation signal and the parameters of the synthesis filter, either on a sample-by-sample or frame-by-frame basis. In other words, the task of the encoder is simply to identify the model parameters of this source-filter model as represented by the decoder structure.

17.2.3 Basic Principle of Analysis by Synthesis

In conventional predictive speech waveform coders, the identification of such model parameters at the encoder is not performed using the analysis-by-synthesis method as defined above by *Bell* et al. in their 1961 paper [17.12]. The encoder in Fig. 17.1a identifies the excitation signal by first calculating the closed-loop prediction residual signal and then directly quantizing this residual signal sample by sample to get the quantized residual (or excitation) signal. The encoder does not perform a synthesis operation when attempting to analyze one of the model parameters, namely, the excitation signal.

In an analysis-by-synthesis speech waveform coder, on the other hand, to achieve better coding efficiency, the prediction residual signal, or excitation signal, is usually quantized block by block rather than sample by sample, where each block is typically 0.5–10 ms long (4–80 samples at 8 kHz sampling). Each block of samples is often called a *vector*. Let the dimension of the excitation vector be K samples and let the excitation encoding bitrate be r bits per sample, then each excitation vector will be represented by Kr bits. Therefore, each vector of the excitation signal can only be quantized into one of 2^{Kr} possible candidate excitation vectors.

An analysis-by-synthesis speech waveform coder does not directly quantize the prediction residual signal as in Fig. 17.1a. Instead, each of the finite 2^{Kr} possible candidates for the excitation signal is passed through the synthesis filter and the resulting synthesized speech signal is compared with the input speech signal. The candidate excitation signal that minimizes a predeter-

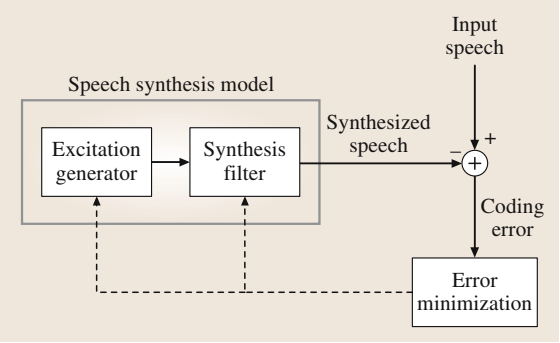


Fig. 17.2 Simplified analysis-by-synthesis speech waveform coder

mined distortion measure between the input speech and the synthesized speech is selected by the encoder as the final excitation signal to be transmitted to the decoder and to be used in the decoder to *excite* the synthesis filter.

The greatly simplified block diagram in Fig. 17.2 illustrates this analysis-by-synthesis coding principle. The gray rectangular box in Fig. 17.2 represents the source-filter speech synthesis model, consisting of an excitation generator followed by a synthesis filter. Note that this speech synthesis model is in direct correspondence to the decoder structure shown in Fig. 17.1b. The difference now is that rather than calculating the prediction residual first and then quantizing it as in Fig. 17.1a, the analysis-by-synthesis coder in Fig. 17.2 passes each of the possible candidate excitation signals through the synthesis filter and identifies the candidate excitation signal that minimizes the error between the corresponding synthesized speech and the input speech. Note that what is shown in Fig. 17.2 is only the encoder structure. The corresponding decoder is simply the excitation generator and synthesis filter enclosed by the gray rectangle.

The encoder in Fig. 17.2 attempts to perform *analysis* of the input speech signal in order to identify the model parameters of the speech synthesis model. The name *analysis-by-synthesis* comes from the fact that the encoder does not try to calculate unquantized model parameters and then quantize them, but instead perform the *analysis* to get the model parameters *by* the *synthesis* of candidate output speech signals using all possible model parameters and identify the set of model parameters that minimizes the error between the synthesized speech and the input speech.

The identification of the best candidate excitation signal that minimizes the coding error is indicated by the dashed arrow going from the *error minimization* block to the *excitation generator* block. This analysis-by-synthesis method of identifying the best model parameters is often called *closed-loop* optimization or quantization, which should not be confused with closed-loop prediction. Closed-loop prediction means the predictor uses previously quantized signal to perform prediction, whereas closed-loop quantization in the current context means the quantization is performed in a way that minimizes the distortion in the speech domain rather than the domain of the parameter to be quantized. In contrast, open-loop quantization means deriving or extracting a parameter and then directly quantize that parameter by minimizing the quantization distortion in the parameter domain.

Strictly speaking, a true analysis-by-synthesis coder will synthesize all possible candidate output speech signals using all combinations of the candidate excitation signals and the candidate synthesis filters and then identify the particular combination that gives the minimum coding error. However, in practice this is rarely done due to the impractically high search complexity that is required. The common approach is to derive the synthesis filter coefficients directly from the input speech signal and perform analysis-by-synthesis coding only for the excitation signal. This approach reduces the performance very little.

17.2.4 Generic Analysis-by-Synthesis Encoder Structure

Figure 17.3 shows a more detailed and more generic encoder structure of an analysis-by-synthesis speech waveform coder. This figure covers most of the analysis-by-synthesis speech coders, at least in the conceptual level. The actual encoder structures used in efficient implementations of such coders may differ, but they are

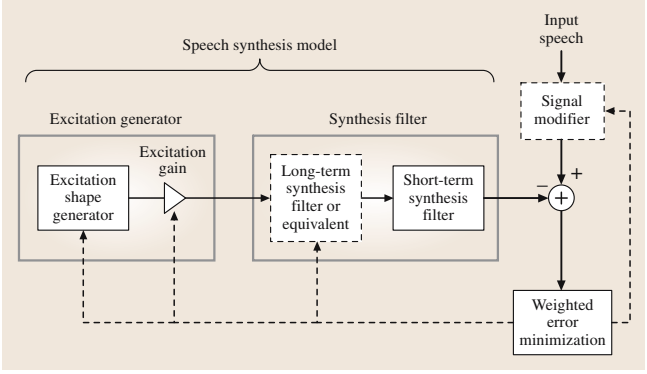


Fig. 17.3 A more detailed generic analysis-by-synthesis speech waveform coder

usually mathematically equivalent to the encoder structure shown in Fig. 17.3. Compared with Fig. 17.2, this figure has

1. added an optional signal modifier for the input speech signal
2. used weighted error minimization
3. expanded the excitation generator and the synthesis filter into more blocks

Each of these differences is briefly explained below.

The addition of the signal modifier is optional; that is why the border of this block is in dashed lines. The purpose of this signal modifier is to modify the input signal in such a way that does not degrade the speech quality appreciably and yet makes the resulting modified speech signal easier for the analysis-by-synthesis coder to encode (i. e., requiring a lower bit rate to encode the signal to the same perceived speech quality) [17.15]. This technique is used to achieve better coding efficiency in some of the CELP coders, such as RCELP [17.15]. However, most analysis-by-synthesis speech coders do not use it because it adds complexity and may occasionally cause slightly audible degradation to speech quality.

The weighted error minimization block in Fig. 17.3 is typically used to shape the coding noise spectrum so that it follows the spectrum of the input signal to some extent – a process usually referred to as *noise spectral shaping* [17.16]. Due to the noise masking effect of the human auditory system, such spectrally shaped coding noise is less audible to human ears.

The excitation generator in Fig. 17.2 is expanded into the excitation shape generator and the excitation gain scaling unit in Fig. 17.3. The excitation shape generator generates excitation vectors with all kinds of possible vector shapes while having gains (as measured by a vector *norm*) either equal to a single value or lie within a narrow range around a single value. The excitation gain scaling unit then scales the excitation shape vectors so that the scaled excitation vector can have a wide dynamic range in terms of vector norm. The separation into generator and scaling blocks is not theoretically necessary, since it is possible for a single excitation generator block to generate excitation signals with all possible shapes and gains. Instead, such separation is motivated by the desire to keep the computational complexity low, since without it a very large excitation codebook is required.

The synthesis filter in Fig. 17.2 is expanded into the cascade of the long-term synthesis filter (or equivalent) and the short-term synthesis filter in Fig. 17.3. A long-term synthesis filter often contains a long-term predictor [17.16] in a feedback loop, and a short-term synthesis filter often contains a short-term predictor in a feedback loop [17.14]. Again, this separation is not theoretically necessary. However, there is an important reason to such decoupling – the long-term synthesis filter (or its equivalent) can model well the quasi-periodicity in voiced speech signals introduced by the periodic vibration of the human vocal cord, while the short-term synthesis filter can model the spectral envelope of speech signals as controlled by the human vocal tract. Thus, such decoupling is motivated by the way the speech signal is produced by a human talker.

Note that the long-term synthesis filter block in Fig. 17.3 is enclosed by dashed lines, signifying that this filter may or may not be used. Most analysis-by-synthesis speech coders do employ this long-term synthesis filter (or its equivalent) as well as the short-term synthesis filter. The G.728 LD-CELP coder [17.6] and the original MPLPC coder [17.2] are two exceptions.

Conceptually, on a block-by-block basis, the encoder in Fig. 17.3 attempts to find the best combination of the excitation shape vector, the excitation gain, the long-term synthesis filter parameters, the short-term synthesis filter parameters, and the signal modifier parameters (if the modifier is present), such that the resulting synthesized speech signal is closest to the modified input speech signal, where *closest* is in the sense of minimizing a weighted mean squared error criterion (distortion measure). Rather than exhaustively searching through all combinations of these parameters, practical analysis-by-synthesis speech coders usually use sequential optimization. In other words, these five sets of parameter values (or four if the signal modifier is not used) are usually optimized one set at a time in a sequential manner so that the computational complexity is manageable. In some speech coders, certain long-term synthesis filter parameters are jointly optimized with at least some of the excitation parameters in an attempt to achieve better coding efficiency. For example, see [17.17].

Essentially all practical analysis-by-synthesis speech coders quantize the short-term synthesis filter parameters in an *open-loop* manner. In other words, the unquantized short-term synthesis filter parameters are first derived from the input speech signal, and then they are directly quantized using a distortion measure in the filter parameter domain rather than using the weighted error in the speech domain as depicted in Fig. 17.3. That is why there is not a dashed arrow from the weight error minimization block to the short-term synthesis filter block in Fig. 17.3.

Exactly how many parameter sets should be closed-loop or jointly quantized depends on the coder design goals and requirements. In some analysis-by-synthesis speech coders that emphasize low computational complexity, only the excitation shape is closed-loop quantized and all other parameter sets are open-loop quantized. As long as the encoding bit rate is not overly low and the speech coder is carefully designed, even such an analysis-by-synthesis coder can achieve fairly high speech quality.

17.2.5 Reasons for the Coding Efficiency of Analysis by Synthesis

As mentioned earlier, the analysis-by-synthesis excitation coding is one of the most efficient speech coding techniques and is the main driving force behind the 4 to 8 times reduction in speech coder bit-rate in the last two decades. A fundamental question to ask is: why is this analysis-by-synthesis technique for predictive speech coders so powerful? One reason is that the speech synthesis model in Fig. 17.3 is a good fit to how speech signal is produced by a human talker.

Perhaps a more important reason is the analysis-by-synthesis method itself. Think of it this way: the task of the encoder is to find the quantized parameters of this speech synthesis model so that the synthesized output speech sounds closest to the input speech. Given this, then which other coding method is better than trying all possible quantized parameters and see which one gives an output speech signal closest to the input speech signal? This *trying all possible values and finding which one gives the best output speech* is the essence of analysis-by-synthesis coding. It is a stark contrast to earlier coding methods where the unquantized optimal parameter value is first derived and then directly quantized using a distortion measure in that parameter domain.

Yet a third reason is that such an analysis-by-synthesis approach naturally lends itself to enable the use of the so-called *vector quantization* (VQ) , which has a higher coding efficiency than conventional sample-by-sample *scalar quantization* as used in ADPCM. Some might argue that ADPCM and adaptive predictive coding (APC) [17.16] are analysis-by-synthesis coders since minimizing the quantization error in the prediction residual domain is mathematically equivalent to minimizing the coding error in the speech domain. While there is indeed such a mathematical equivalence in the degenerate case of scalar quantization, it is questionable, in our opinion, whether ADPCM and APC should be called analysis-by-synthesis coders. Our main objection is that these scalar-quantization-based coders never really perform the *synthesis* operation when trying to do the *analysis* of the model parameters (i. e., when performing residual quantization).

Conventional ADPCM and APC coders are restricted to using sample-by-sample scalar quantization and cannot be extended to vector quantization to reap the benefit of the higher coding efficiency of VQ. (If these coders indeed use VQ, then they will not be called ADPCM and APC anymore and will be called CELP.) These coders need to compute the unquantized prediction residual signal first and then quantize it, but with VQ the unquantized prediction residual signal depends on the quantized prediction residual signal due to the feedback filter structure in Fig. 17.1a. This creates a chicken-and-egg problem. In contrast, the analysis-by-synthesis approach completely avoid this chicken-and-egg problem by directly trying all possible quantized residual (excitation) signals without the need to compute the unquantized residual signal first. Thus, by enabling the use of VQ for the excitation signal, the analysis-by-synthesis approach reaps the benefit of the higher coding efficiency of VQ.

This chapter only explains the most fundamental basic ideas of analysis-by-synthesis coding. If the encoder structure in Fig. 17.3 were implemented as is, the resulting complexity would be quite high. In later sections, computationally much more efficient but mathematically equivalent encoder structures will be introduced. Most analysis-by-synthesis speech coders today are implemented based on the more efficient encoder structures.

17.3 Overview of Prominent Analysis-by-Synthesis Speech Coders

This section gives an overview of some prominent analysis-by-synthesis speech coders and shows the relationship between them in the form of a *family tree*. There are numerous analysis-by-synthesis speech coders proposed in the literature. It is not practical to describe all of them in this chapter. Instead, the intention here is to describe only those analysis-by-synthesis speech coders that are either standard coders or are the first of its kind and thus are definitive and seminal.

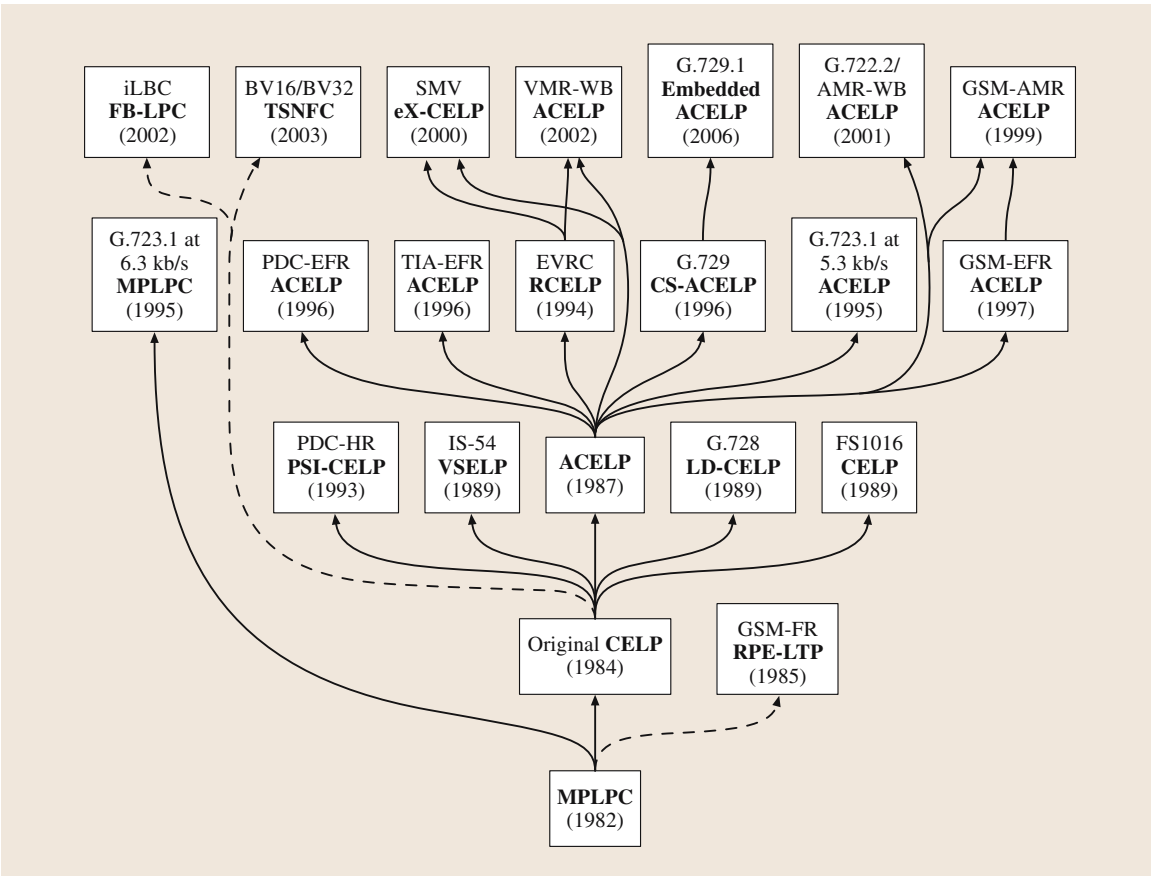


Fig. 17.4 Family tree of prominent analysis-by-synthesis speech waveform coders

Figure 17.4 shows the family tree of these selected analysis-by-synthesis speech coders. Each rectangular block in the tree represents a particular speech coder or a type of speech coders. Within each rectangular block, the first row (sometimes the first two rows) specifies the name of the coding standard, the second row in boldface letters gives the name of the coding technique, and the third row provides the year of the first publication the authors can find for that coder (either as a technical paper or as a standard specification).

The speech coders in Fig. 17.4 are organized in five different rows, with each row roughly corresponding to a different generation of coders. Each generation represents a group of analysis-by-synthesis speech coders developed within a time period of a few years.

The MPLPC coder [17.2] proposed in 1982 is the first modern analysis-by-synthesis speech waveform coder. The original CELP coder [17.3] and the RPE-LTP coder [17.18] in the second row represent the second-generation analysis-by-synthesis coders developed in the mid 1980s. The five coders in the third row represent the third-generation analysis-by-synthesis coders developed in the late 1980s and early 1990s. The seven coders in the fourth row represent the fourth-generation analysis-by-synthesis coders developed in the mid 1990s. Finally, the seven coders in the fifth row represent the fifth-generation analysis-by-synthesis coders developed from the late 1990s to the present day. Strictly speaking, the distinctions between the fourth- and fifth-generation coders are not always clear; however, at least the fifth-generation coders were developed later than the fourth-generation coders.

A brief overview of this family tree of analysis-by-synthesis speech coders is given below. The MPLPC coder proposed in 1982 is at the root of the family tree, since it can be argued that all other coders in Fig. 17.4 can trace their roots back to this coder.

For the second-generation coders, the original CELP coder proposed in 1984 is a direct descendant of MPLPC with long-term prediction [17.19], as *Atal* worked on both coders and the two coders are similar except the multipulse model is replaced by VQ of the excitation in CELP. The excitation VQ codebook of this CELP coder is populated by Gaussian random numbers. The Groupe Spèciale Mobile (GSM) full-rate standard coder regular-pulse excitation with long-term prediction (RPE-LTP) [17.18] is inspired by the MPLPC coder; however, to reduce the computational complexity, it uses an excitation model of regularly spaced pulses which deviates substantially from the original multipulse model. That is why the arrow from MPLPC to RPE-LTP has a dashed line.

The third-generation coders saw the most variety of analysis-by-synthesis excitation coding techniques. The first standard coder based on CELP is the US federal standard FS1016 CELP coder at 4.8 kb/s [17.20]. It uses overlapping center-clipped Gaussian VQ codevectors for excitation coding. The second CELP coder that became a standard coder is the Telecommunications Industry Association (TIA) IS-54 standard vector sum excited linear prediction (VSELP) coder at 8 kb/s [17.17]. It uses sums of several basis vectors as the excitation VQ codevectors. A 6.7 kb/s VSELP coder later also became the Japanese personal digital cellular (PDC) full-rate standard coder, and a 5.6 kb/s VSELP coder became the GSM half-rate standard coder [17.21]. The third CELP coder that became a standard is the ITU-T G.728 standard low-delay CELP (LD-CELP) coder at 16 kb/s [17.6]. It uses a closed-loop trained-gain-shape VQ codebook structure with a small vector dimension and with joint optimization of VQ gain and shape. The fourth CELP coder that became a standard is the Japanese PDC half-rate standard pitch synchronous innovation CELP (PSI-CELP) coder at 3.45 kb/s [17.22]. This coder uses a fairly long vector dimension and repeats the first portion of the VQ codevectors in a pitch-synchronous manner if the pitch period is less than the vector dimension.

An influential class of third-generation coder is the algebraic CELP, or ACELP [17.23]. This coder reduces the VQ codebook search complexity by using an algebraic VQ codebook, with the elements of VQ codevectors having only $+1, 0$, and -1 as the possible sample values. As can be seen from Fig. 17.4, most of the fourth- and fifth-generation coders are derivatives of ACELP. However, it should be noted that ACELP-based speech coding standards use sparse algebraic codes [17.24] and not the original algebraic code proposed in [17.23]. The sparseness contributes to a low computational complexity and the reduction of granular type of coding noise.

All but one fourth-generation standard coders in Fig. 17.4 can trace their roots to ACELP. The single exception is the 6.3 kb/s version of the ITU-T G.723.1 standard [17.25]. It is an improved version of the first-generation MPLPC coder. One of the fourth-generation coders that deserves attention is the so-called relaxed CELP (RCELP) coder used in North American cellular standard enhanced variable-rate coder (EVRC) [17.26]. It is based on a generalization of analysis-by-synthesis [17.15] that adaptively modifies the input speech signal in such a way that the modified signal sounds essentially the same as the original input signal and yet it becomes easier for a CELP coder to encode the signal efficiently. This RCELP technique is also used in the North American cellular standard selectable mode vocoder (SMV) [17.27] and variable-rate multimode wide-band (VMR-WB) coder [17.28].

For fifth-generation standard coders in Fig. 17.4, all but two are based on ACELP. The exceptions are the BV16/BV32 coder [17.29], [17.30] and the internet low bit rate (iLBC) coder [17.31], [17.32]. The iLBC coder is an Internet Engineering Task Force (IETF) experimental standard and also a PacketCable standard. It is based on what is called forward backward linear predictive coding (FB-LPC), where a maximum-energy segment of the short-term prediction residual is independently coded, and then a dynamic codebook is constructed first forward in time and then backward in time. The 16 kb/s BroadVoice16 (BV16) coder [17.33] and its wide-band version 32 kb/s BroadVoice32 (BV32) coder use similar coding algorithms. BV16 is a PacketCable standard, SCTE (Society of Cable Telecommunications Engineers) standard, and American National Standards Institute (ANSI) standard for voice-over-internet protocol (VoIP) applications in cable telephony, and BV32 is a PacketCable 2.0 standard. The BV16/32 coder is based on two-stage noise feedback coding (TSNFC). This coder is not a CELP coder since its encoder structure is totally different from that of CELP; however, it employs many of the complexity reduction and even codebook design techniques of CELP within the TSNFC encoder structure. For this reason, the arrow from the original CELP to TSNFC has a dashed line, signifying that it is not a direct descendant of CELP but leverages nearly two decades of research on CELP.

As discussed above, recent speech coding standards are dominated by ACELP-related coders. For this

reason, ACELP and its derivatives will receive more-thorough treatment in this chapter. Nevertheless, there are still many other interesting kinds of analysis-by-synthesis coders in this family tree. In the sections that follow, the excitation coding methods for the more distinctive types of coders in this family tree will be described in more detail. Due to the similarity between many different ACELP coders in Fig. 17.4, many of them are lumped together and discussed in the same ACELP section.

17.4 Multipulse Linear Predictive Coding (MPLPC)

The MPLPC coder [17.2] was the first predictive waveform coder to abandon the sample-by-sample residual quantization procedure and to encode the entire block of excitation signal as a single entity in an analysis-by-synthesis manner. It achieves this by using an excitation signal with mostly zero samples and finding the optimal locations and amplitudes for a small number of nonzero pulses such that the resulting synthesized speech signal minimizes a weighted distortion measure in the speech domain.

Refer to the generic analysis-by-synthesis speech waveform coder shown in Fig. 17.3. The MPLPC coder does not use the signal modifier in Fig. 17.3. The original MPLPC coder proposed in [17.2] also does not use the long-term synthesis filter. In fact, this original MPLPC coder was developed to improve the linear predictive coding (LPC) vocoder [17.34], which does not use a long-term synthesis filter and have a rigid excitation model of either periodic pulse train or white noise.

The MPLPC coder does use the short-term synthesis filter in Fig. 17.3. This short-term synthesis filter is in the form of an all-pole filter given by the following transfer function

$$H_s(z) = \frac{1}{A(z)} , \tag{17.1}$$

where

$$A(z) = 1 - P(z) = 1 - \sum_{i=1}^{M} a_i z^{-i} \tag{17.2}$$

is the short-term prediction error filter, $P(z)$ is the short-term predictor, $a_i, i = 1, 2, \ldots, M$, are the predictor coefficients, and M is the order of the short-term synthesis filter. The filter order M is typically somewhere between 10 and 16 for 8 kHz sampled signals. The predictor coefficients are adaptive and are usually transmitted once every 10–20 ms.

In this first MPLPC coder the weighted error minimization in Fig. 17.3 is achieved through the use of a so-called perceptual weighting filter in the form of

$$W(z) = \frac{A(z)}{A(z/\gamma)} , \quad 0 < \gamma < 1 , \tag{17.3}$$

where

$$A\left(\frac{z}{\gamma}\right) = 1 - P\left(\frac{z}{\gamma}\right) = 1 - \sum_{i=1}^{M} a_i \gamma^i z^{-i} \tag{17.4}$$

is a modified version of $A(z)$ with the roots of the polynomial moved radially toward the origin (the radii of the roots are scaled by a factor of γ, which is typically 0.8). Since the roots are normally within the unit circle for a stable synthesis filter $H_s(z) = 1/A(z)$, the net effect is that the magnitude frequency response of the filter $1/A(z/\gamma)$ is a smoothed version of that of the filter $1/A(z)$.

For MPLPC, the weighted error minimization in Fig. 17.3 is achieved by passing the difference between the input speech signal and the synthesized speech signal through the perceptual weighting filter $W(z)$ and then identifying which combination of multipulse model parameters gives the minimum mean-squared error of this perceptually weighted difference signal. The perceptual weighting filter emphasizes the spectral valley regions and deemphasize the spectral peak regions of the input speech spectrum. This has an effect of allowing more coding noise under the spectral peaks and suppressing coding noise in the spectral valleys of speech. Thus, the perceptual weighting filter shapes the coding noise spectrum so that it follows the input speech spectrum to some extent. Due to the noise masking effect of the human auditory system, such a spectrally shaped coding noise is perceptually less audible than coding noise with a flat spectrum.

The short-term synthesis filter $H_s(z) = 1/A(z)$ has a magnitude frequency response corresponding to the spectral envelope of the input signal. Thus, the filter $H_s(z/\gamma) = 1/A(z/\gamma)$ has a magnitude response corresponding to a smoothed version of the spectral envelope of input speech. Subtracting one magnitude response

from another in the logarithmic domain is equivalent to dividing in the linear domain. Hence, the filter

$$\frac{1}{W(z)} = \frac{A(z/\gamma)}{A(z)} = \frac{H_s(z)}{H_s(z/\gamma)} \tag{17.5}$$

will have a magnitude response somewhat similar to the spectral envelope of the input speech but with a reduced spectral slope. Since this filter is the inverse of the perceptual weighting filter $W(z)$, given how the weighted error minimization works as described above, the magnitude response of this filter $1/W(z)$ is the spectral envelope of the coding noise that the MPLPC coder will achieve. For an example of what the magnitude responses of $H_s(z)$ and $W(z)$ look like, see [17.2].

The first MPLPC coder in [17.2] encoded the excitation signal block-by-block in an analysis-by-synthesis manner using a block size of 5 ms, or 40 samples at 8 kHz sampling. Most of the 40 samples are zeroes. Only about 10% of the 40 excitation samples (four samples) have nonzero amplitudes. These nonzero samples are called *pulses* because they look like pulses in the graphical representation of such a digital excitation signal. Since there are usually multiple of such nonzero samples in the block of excitation signal to be encoded, the resulting excitation model is called the *multipulse* model, and the resulting linear predictive coding scheme is called *multipulse linear predictive coding* (MPLPC).

Let m be the number of pulses in each block of excitation signal. Each pulse is completely specified by its amplitude and its location within the block. Therefore, the multipulse excitation model has $2m$ model parameters that need to be determined. Jointly optimizing these $2m$ model parameters would cost too high computational complexity. A suboptimal sequential optimization approach is proposed in [17.2].

In the sequential optimization approach, the m pulses are determined one at a time. For a given pulse location, the corresponding optimal amplitude can be obtained through a closed-form solution [17.2]. Before determining the location and amplitude of the first pulse, with the excitation signal set to zero, the output of the short-term synthesis filter during the current block (due to the nonzero filter memory left over at the end of the last block) is subtracted from the input speech. The result is the target signal for the search of the first pulse. Each of the 40 possible pulse locations is tried. With closed-loop optimal amplitude solved for each pulse location, the pulse is passed through the short-term synthesis filter. The filter output signal is subtracted from the target signal and the resulting difference signal is passed through the perceptual weighting filter. The mean-squared error (MSE) of the weighted signal is calculated. This process is repeated until the best pulse location and amplitude that gives the lowest weighted error is identified. Next, the short-term synthesis filter output signal due to this first pulse is subtracted from the target signal to form a new target signal for the search of the second pulse. This process is repeated until the pulse locations of all m pulses are determined. Then, given the m pulse locations, the m pulse amplitudes can be jointly optimized in a single step [17.2].

Using this approach, it is reported in [17.2] that only minimal audio quality improvement are achieved after about eight pulses have been placed in a 10 ms interval. It is also reported [17.2] that the resulting output speech quality sounded natural and perceptually close to the input speech, without the common unnatural and buzzy characteristics of the LPC vocoder output speech.

Strictly speaking, since each pulse in the multipulse model is scaled by a different amplitude, the decomposition of the excitation generator into the excitation shape generator and the excitation gain in Fig. 17.3 is not a good description. This problem can be avoided if the excitation gain in Fig. 17.3 is understood to have the possibility of taking the form of a gain vector (with dimension m), where each element of the gain vector is individually used to scale one of the m pulses.

Further improvements to this initial MPLPC coder was proposed in [17.19]. The most notable is the addition of a pitch predictor, or equivalently, the addition of the long-term synthesis filter in Fig. 17.3. Due to the quasiperiodicity in voiced speech such as vowels, it is found that the multipulse excitation signal shows significant correlation from one pitch period to the next. This correlation can be exploited by using a pitch predictor (also called long-term predictor). Let the pitch period be T samples, and let β be the long-term predictor coefficient. Then, with

$$H_l(z) = \frac{1}{1-\beta z^{-T}} \tag{17.6}$$

chosen as the transfer function of the long-term synthesis filter in Fig. 17.3, each pulse at the output of the long-term synthesis filter will be scaled by β and then be added to the excitation signal T samples later. This has the effect of creating a periodic pulse pattern at a period of T samples. After adding this long-term synthesis filter, fewer pulses are needed to produce the same level of speech quality.

Another improvement proposed in [17.19] is related to pulse amplitude optimization. Rather than waiting until all pulse locations are determined and then jointly

optimize all the pulse amplitudes, the proposed procedure performs reoptimization of amplitudes along the way. In other words, during the sequential search for pulse locations, after the j-th pulse location is determined ($1 \le j \le m$), all pulse amplitudes from the first pulse to the j-th pulse are reoptimized jointly before continuing to search for the $(j+1)$-th pulse location. It is reported [17.19] that this approach improved coder output signal-to-noise ratio (SNR) by 2–10 dB, with an average of slightly over 3 dB.

There are many other improvements and variations proposed in the literature for the MPLPC coder. Due to the space limitation, they have to be omitted in the discussion here.

The multipulse excitation model is used in the ITU-T Recommendation G.723.1 speech coder. G.723.1 has two bit-rates. The lower-bit-rate 5.3 kb/s version is based on ACELP, while the higher-bit-rate 6.3 kb/s version is based on MPLPC with long-term prediction.

This 6.3 kb/s version of G.723.1 uses a frame size of 30 ms, which is equally divided into four subframes of 7.5 ms (60 samples) each. Multipulse excitation search is performed on each of the four subframes. The multipulse excitation model uses six pulses for even subframes and five pulses for odd subframes. The pulse location is restricted to be either all even or all odd, as indicated by a grid bit. Furthermore, all pulses are constrained to have the same magnitude, although different pulses can have different signs. The shared magnitude is first estimated and quantized. Then, four quantized magnitude values around the estimated magnitude are allowed in the analysis-by-synthesis multipulse search. For each of these four possible quantized magnitude values, the pulse locations and signs are sequentially optimized one pulse at a time. This procedure is repeated for both the even and odd pulse position grids. Finally, the best combination of all these parameters (quantized magnitude, pulse locations and signs, and even or odd grid) that gives the minimum weighted error of the synthesized speech is selected as the final multipulse excitation parameters to be transmitted to the G.723.1 decoder.

As can be seen, this G.723.1 multipulse coder has deviated substantially from the original MPLPC proposed in [17.2] and [17.19]. In fact, with all the constraints put on the G.723.1 multipulse model (due to the low-bit-rate limitation), the G.723.1 multipulse excitation signal starts to resemble to some extent the excitation signal of ACELP coders. However, such constraints aside, the spirit of the original multipulse excitation model can still be seen in this 6.3 kb/s G.723.1 multipulse coder.

17.5 Regular-Pulse Excitation with Long-Term Prediction (RPE-LTP)

The regular-pulse excitation (RPE) coder [17.35] can be regarded as a special low-complexity realization of the fundamental analysis-by-synthesis concept proposed in the original multipulse LPC coder [17.2]. A special version of it, the regular-pulse excitation with long-term prediction (RPE-LTP) coder at 13 kb/s [17.18], was selected as the first GSM standard coder for European digital cellular service.

The RPE excitation model consists of J possible sequences of regularly spaced pulses, each with the pulses located at a different phase. The typical value of J is 3 or 4. As an example, for the k-th sequence of pulses, the nonzero excitation samples (the pulses) are located at the positions of $k, k+J, k+2J, k+3J, \ldots$, and the sample values at other locations are zero. The task of excitation coding is to find the amplitude values for each of the pulses in each of the J sequences of regularly spaced pulses, and then find the sequence of pulses that minimizes a weighted error measure.

Similar to the original MPLPC coder proposed in [17.2], the initial RPE coder [17.35] also did not use a long-term predictor. On a conceptual level, the encoder structure of the initial RPE coder is equivalent to the structure in Fig. 17.3 without the long-term synthesis filter and the signal modifier. However, in actual implementation, the LPC inverse filter $A(z)$, which is the numerator portion of the perceptual weighting filter $W(z) = A(z)/A(z/\gamma)$, is moved beyond the adder to the left and above. The one to the left of the adder cancels out the short-term synthesis filter, while the one above the adder stays there and filters the input speech to produce the short-term prediction residual. The remaining denominator portion of the weighting filter, or $1/A(z/\gamma)$, stays below the adder for weighted error minimization.

In [17.35], the procedure for the RPE analysis-by-synthesis excitation search is given. It basically amounts to solving J sets of linear equations to find the optimal amplitude for each pulse of the J possible regular pulse sequences. The weighted distortion measure for each of the J pulse sequences are calculated and the sequence that minimizes the weighted distortion is selected as the final excitation sequence.

In the GSM full-rate (GSM-FR) standard RPE-LTP coder, a single-tap long-term predictor (LTP) is added to the initial RPE coder, so GSM-FR corresponds to the coder structure in Fig. 17.3 with the long-term synthesis filter enabled. The GSM-FR coder uses three possible sequences of regularly space pulses, that is, $J = 3$. It has a frame size of 20 ms and a subframe size of 5 ms. Therefore, for every 5 ms subframe of 40 samples, there are 3 possible sequences of regularly spaced pulses. Each sequence contains only 13 or 14 nonzero pulses, with the remaining samples having zero amplitudes. The selected sequence is identified by 2 bits, and the corresponding pulse amplitudes are encoded with a 3-bit block-adaptive PCM quantizer. The block maximum for each subframe is encoded with 6 bits. The pitch period and the pitch tap are derived once a subframe and quantized to 7 bits and 2 bits, respectively.

This GSM-FR RPE-LTP coder is the speech coder for the first-generation GSM digital cellular telephones. It became the first analysis-by-synthesis coder that was standardized and widely deployed across many countries.

17.6 The Original Code Excited Linear Prediction (CELP) Coder

By using an analysis-by-synthesis procedure, the multipulse excitation model in the MPLPC coder [17.2] opened the door for significantly more efficient excitation coding than was possible with earlier speech waveform coders. However, even with this multipulse model, there is still a limit on how low the excitation encoding bit-rate can go. For example, with the same 40-sample excitation block size and four pulses as in the original MPLPC, suppose it is desirable to encode the excitation signal at a bit rate of 1/4 bits/sample, this gives a mere 10 bits to represent the four pulse locations and four pulse amplitudes. It is virtually impossible to use 10 bits to quantize such eight parameters with sufficient accuracy. This problem led *Atal* and *Schroeder* to use a 10 bit vector quantization (VQ) codebook to quantize the excitation signal [17.3], leading to what is known today as code-excited linear prediction, or CELP [17.4].

To use 10-bit VQ to quantize 40 samples of the excitation signal, Atal and Schroeder constructed a VQ codebook containing $2^{10} = 1024$ codevectors, each of which is a 40-dimensional vector containing 40 white Gaussian random numbers with unit variance. Beside conceding that it is generally difficult to design an optimum deterministic codebook (with a total of 40 960 samples in it), in [17.4] *Schroeder* and *Atal* actually gave a justification for using the Gaussian random numbers. The justification is that the probability density function of the prediction residual signal after both short- and long-term prediction is nearly Gaussian.

The encoder structure of this initial CELP coder fits the structure shown in Fig. 17.3 exactly. Again, the signal modifier in Fig. 17.3 is not used in this CELP coder. The excitation shape generator is basically a table look-up of the 10-bit, 40-dimensional excitation VQ codebook described earlier. The excitation gain scales the unit-variance Gaussian codevectors to the proper gain level that matches the root mean square (RMS) value of the prediction residual signal after both short-term and long-term prediction. It is updated once every 5 ms together with the excitation codevector.

The synthesis filter of CELP contains both the long-term synthesis filter and the short-term synthesis filter. The short-term synthesis filter has the same form as given in (17.1), while the long-term synthesis filter uses a 3-tap pitch predictor rather than the single-tap pitch predictor given in (17.6). The weighted error minimization of CELP is also achieved with a perceptual weighting filter in exactly the same way as in MPLPC, described earlier in Sect. 17.4. The perceptual weighting filter $W(z)$ also has exactly the same form as defined in (17.3).

Atal and *Schroeder* reported [17.3, 4] that when the 40-sample excitation vector is quantized to 0.25 bit/sample in an analysis-by-synthesis manner using a 10-bit Gaussian VQ codebook, with all other speech synthesis model parameters (excitation gain and parameters for the long- and short-term synthesis filters) left at their open-loop-derived optimal values, the resulting synthesized speech sounded very close to the original input speech. Only small differences were noticeable even in close pairwise comparisons over headphones.

Achieving such a high level of output speech quality with such a low bit rate of merely 0.25 bit/sample for the excitation signal was considered a stunning breakthrough in 1984–1985. Almost immediately, a large number of speech coding researchers jumped in and engaged in the research of the CELP coding technique. In the subsequent years that followed, hundreds of CELP-related technical papers have been published and numerous advancements in CELP coding have been

made. As shown in the family tree of Fig. 17.4, this analysis-by-synthesis CELP coding idea eventually led to more than a dozen of modern low-bit-rate speech coding standards, and CELP-related coders dominate almost all speech coding standards established since the late 1980s.

The two initial CELP papers [17.3, 4] were more *proof of concept* papers than actual coder design papers. Not only were all model parameters except the excitation signal were left unquantized, but a major issue was the very high computational complexity required by the analysis-by-synthesis excitation VQ codebook search.

Actually, it is fairly easy to estimate the complexity of such a codebook search. Refer to Fig. 17.3. Given that the short-term filter order they used was $M = 16$, each of the 1024 excitation codevectors has to be scaled by the excitation gain (one multiply per sample), filtered by the long-term synthesis filter (three multiply-adds per sample), filtered by the short-term synthesis filter (16 multiply–adds per sample), subtracted from the input speech (one subtract per sample), filtered by the perceptual weighting filter ($2 \times 16 = 32$ multiply–adds per sample), squared (one multiply per sample), and then added (one add per sample) to get the weighted distortion value of that codevector. Therefore, at a sampling rate of 8000 Hz, the total complexity for searching through all 1024 excitation codevectors to identify the one that minimizes the weighted distortion measure would take at least $8000 \times 1024 \times (1+3+16+1+32+1+1) = 450\,560\,000$ operations per second, or 450.6 MFLOPS (million floating-point operations per second). Even the fastest supercomputer at that time could not perform computations that fast. No wonder *Atal* and *Schroeder* reported [17.3, 4] that their initial CELP simulation took 125 s of central processor unit (CPU) time on the then-supercomputer Cray-1 to process just one second of speech.

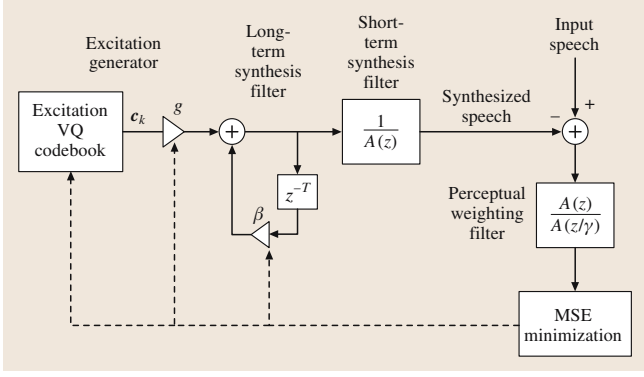


Fig. 17.5 Encoder structure of the original CELP coder

However, even *Atal* and *Schroeder* pointed out [17.4] at that time that

> *a code book with sufficient structure amenable to fast search algorithms could lead to real-time implementation of code-excited coders.*

This turned out to be true. In subsequent years many specially structured excitation VQ codebooks that allowed fast codebook search were proposed – the excitation codebooks of ACELP, VSELP, and FS1016 CELP are all good examples.

However, even without such specially structured excitation codebooks, by just re-arranging the encoder structure in Fig. 17.3 and performing the CELP excitation codebook search in a mathematically equivalent way, the computational complexity can be reduced by almost an order of magnitude. This efficient encoder structure and excitation codebook search procedure is explained below. It forms the basis of essentially all practical CELP-based coders today.

Consider the encoder structure shown in Fig. 17.5. This structure is basically the same as the generic analysis-by-synthesis coder structure shown in Fig. 17.3, except that specific filter structure and transfer functions are given. For convenience of later discussion of the so-called *adaptive codebook*, the long-term synthesis filter is reverted from a three-tap filter back to a single-tap filter as used in the improved MPLPC [17.19]. The short-term synthesis filter is $1/A(z)$. The weighted error minimization is explicitly separated into a perceptual weighting filter as defined in (17.3) followed by MSE minimization. Even with the three-tap pitch filter replaced by a single-tap one, the complexity is still $8000 \times 1024 \times (1+1+16+1+32+1+1) = 434.2$ MFLOPS.

As suggested in [17.19], the perceptual weighting filter can be moved before the adder in Fig. 17.5 so that the input speech and the synthesized speech are each individually weighted before the difference of the two weighted signals are calculated. The cascade of the short-term synthesis filter and the perceptual weighting filter gives a weighted short-term synthesis filter in the form of

$$H(z) = \frac{W(z)}{A(z)} = \frac{1}{A(z)} \frac{A(z)}{A(z/\gamma)} = \frac{1}{A(z/\gamma)} \,. \qquad (17.7)$$

This is shown in Fig. 17.6. Even this step alone can cut the computational complexity by almost a factor of three. There is more that can be saved. However, before

continuing, let us briefly describe another new feature in Fig. 17.6 – the concept of the adaptive codebook as introduced in [17.36].

The adaptive codebook in Fig. 17.6 is a different way to achieve a similar effect as the long-term synthesis filter in Fig. 17.5. If the pitch period T is not smaller than the excitation vector dimension, then once the effect of the filter memory is taken out, the long-term synthesis filter in Fig. 17.5 does not affect the excitation codebook search at all due to the bulk delay z^{-T}. However, if the pitch period T is smaller than the vector dimension, then different excitation VQ codevectors will cause different contributions from the long-term synthesis filter to be added to the excitation to the short-term synthesis filter. This makes it more difficult to perform certain efficient codebook search methods.

To facilitate an efficient codebook search, the effect of the long-term synthesis filter in Fig. 17.5 is modeled by the adaptive codebook in Fig. 17.6. If the pitch period is not smaller than the excitation vector dimension, then the adaptive codebook simply contains different vector sections of the long delay line in the long-term synthesis filter defined by the range of the pitch period. In other words, the codevectors in the adaptive codebook is obtained by using a sliding rectangular window to extract different sections of the long delay line in the long-term synthesis filter, with different codevectors corresponding to different target pitch periods. In this case, the adaptive codebook, the adaptive codebook gain β, and the related adder in Fig. 17.6 is mathematically equivalent to the long-term synthesis filter in Fig. 17.5.

On the other hand, if the pitch period is smaller than the vector dimension, then at least a portion of the codevectors in the adaptive codebook will be beyond the long delay line of the long-term synthesis filter and will correspond to the samples in the current excitation vector that have not been determined yet. Rather than having incomplete codevectors, the adaptive codebook will periodically repeat the samples in the long delay line at the target pitch period.

Thus, the concept of adaptive codebook allows the use of large excitation vector dimension which tends to improve the coding efficiency. In addition, it allows the long-term synthesis filter to be modeled as just another stage of excitation VQ, thus simplifying the codebook search procedure.

The codevectors in an adaptive codebook is changing with time. That is why the codebook is called the *adaptive* codebook. In contrast, the original excitation VQ codebook in Fig. 17.5 does not change with time. Therefore, as shown in Fig. 17.6, it is often called the *fixed* codebook in a CELP coder where an adaptive codebook is used.

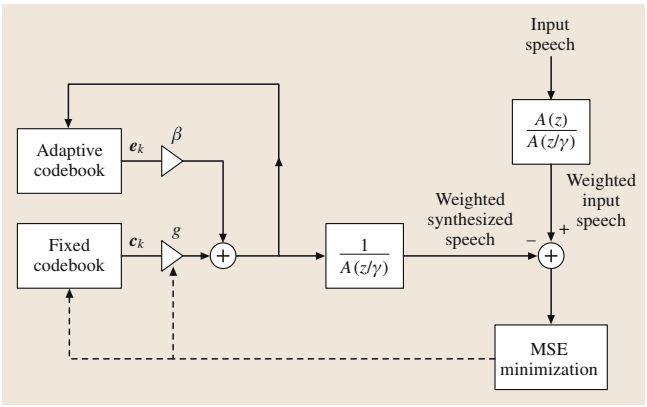


Fig. 17.6 Intermediate encoder structure of the efficient CELP coder

The encoder structure in Fig. 17.6 can be changed to an even more efficient but mathematically equivalent structure as shown in Fig. 17.7. According to the linear system theory, the output of the filter $H(z)$ is the sum of two components

1. the zero-state response (ZSR), which is the output of the filter due to the input signal, with the initial filter memory set to zero,
2. the zero-input response (ZIR), which is the output of the filter due to only the filter memory, with the input signal set to zero.

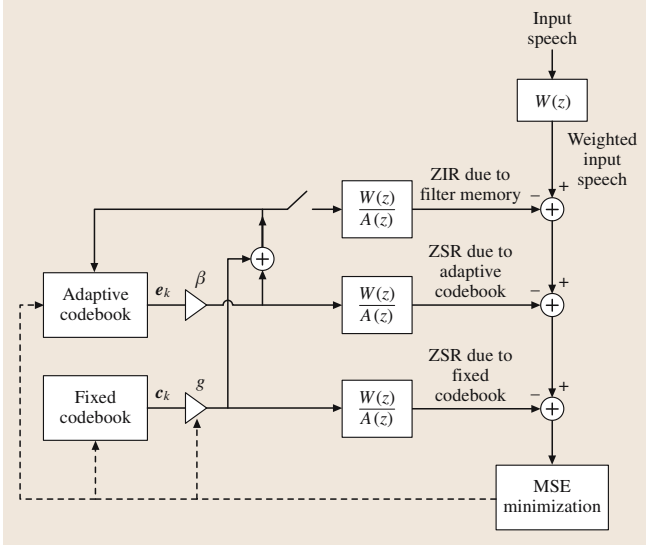


Fig. 17.7 Encoder structure of the efficient CELP coder

Note that the ZIR does not depend on which excitation VQ codevector is used. Thus, by decomposing the output of the weighted short-term synthesis filter $H(z)$ into ZIR and ZSR, the ZIR component can first be subtracted out of the weighted speech signal, and the resulting signal becomes the *target signal* for the codebook search based on the ZSR component. In fact, this idea was already suggested by *Atal* and *Remde* in their original MPLPC paper [17.2], even though they did not use the terms of ZIR and ZSR.

In some later CELP coders, the perceptual weighting filter took a form different from what was specified in (17.3). Therefore, to keep Fig. 17.7 general, the weighted short-term synthesis filter is represented as $H(z) = W(z)/A(z)$.

In Fig. 17.6, the excitation signal to the short-term synthesis filter has two components: the scaled adaptive codebook vector $\beta \boldsymbol{e}_k$ and the scaled fixed codebook vector $g\boldsymbol{c}_k$. Therefore, the weighted synthesized speech signal in Fig. 17.6 can be decomposed into three components: the ZIR signal due to the memory of the filter $W(z)/A(z)$, the ZSR signal due to the adaptive codebook, and the ZSR signal due to the fixed codebook.

In the efficient CELP excitation codebook search method based on Fig. 17.7, the input speech vector is first passed through the perceptual weighting filter $W(z)$ to get the weighted input speech vector. Then, the ZIR vector due to the filter memory is calculated by setting the initial memory of the weighted short-term synthesis filter to the filter memory at the last sample of the last input speech vector and letting the filter *ring* without any input excitation signal (the *switch* in Fig. 17.7 is open). This ZIR vector is subtracted from the weighted input speech vector to get the target vector for the next stage. Next, the adaptive codebook index (the pitch period) and the adaptive codebook gain (the pitch gain) are usually determined in an analysis-by-synthesis manner to minimize the MSE between the target vector obtained above and the ZSR vector due to the adaptive codebook. The ZSR vector corresponding to the best combination of the pitch period and the pitch gain is then subtracted from the target vector to get the next target vector for the ZSR vector due to the fixed codebook. Finally, the fixed codebook index and the fixed codebook gain are determined in an analysis-by-synthesis manner to minimize the MSE between this next target vector and the ZSR vector due to the fixed codebook. After such a codebook search, the selected codebook vectors and gains are used to calculate the sum of the two excitation components $\beta \boldsymbol{e}_k + g\boldsymbol{c}_k$, and the resulting final excitation vector is used to update the adaptive codebook and the memory of the weighted short-term synthesis filter (the *switch* is closed). Then, this procedure is repeated for the next input speech vector.

The encoder structure in Fig. 17.7 looks much more complicated than the structure in Fig. 17.5, so why is it computationally more efficient? The key lies in the fact that by subtracting out the ZIR vector due to filter memory, the search through the adaptive codebook and the fixed codebook can be performed without the actual filtering operation. Let $h(n), n = 0, 1, \ldots, N-1$ be the truncated impulse response of the weighted short-term synthesis filter $H(z) = W(z)/A(z)$ with N being the dimension (length) of the excitation vector. Then, with the initial filter states (memory) set to zero in the lower two branches in Fig. 17.7 containing the filter $W(z)/A(z)$, the filtering operation in these two branches can be replaced by convolution, which can be represented by a matrix–vector multiplication operation [17.37, 38].

Take the fixed codebook search as an example. Let the target vector for the fixed codebook search be $\boldsymbol{t} = [t(0), t(1), \cdots, t(N-1)]^{\mathrm{T}}$, let the k-th fixed codebook vector be $\boldsymbol{c}_k = [c_k(0), c_k(1), \cdots, c_k(N-1)]^{\mathrm{T}}$, and let

$$\boldsymbol{H} = \begin{pmatrix} h(0) & 0 & \cdots & 0 \\ h(1) & h(0) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ h(N-2) & h(N-3) & \cdots & 0 \\ h(N-1) & h(N-2) & \cdots & h(0) \end{pmatrix} \tag{17.8}$$

be the lower triangular Toeplitz matrix populated by the impulse response of the weighted short-term synthesis filter $H(z) = W(z)/A(z)$. Then, the ZSR vector due to the k-th fixed codebook vector can be represented as $g\boldsymbol{H}\boldsymbol{c}_k$, and the codebook search find the codebook index k that minimizes the MSE distortion measure

$$E_k = \|\boldsymbol{t} - g\boldsymbol{H}\boldsymbol{c}_k\|^2 , \tag{17.9}$$

where $\|\cdot\|^2$ indicates the Euclidean norm, or the sum of squares of the vector components. This distortion measure can be expanded as

$$E_k = \|\boldsymbol{t}\|^2 - 2g\boldsymbol{t}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{c}_k + g^2\|\boldsymbol{H}\boldsymbol{c}_k\|^2 . \tag{17.10}$$

Since the energy of target vector $\|\boldsymbol{t}\|^2$ is independent of the codebook index k, this term can be ignored when minimizing the distortion measure above. Thus, calculating the distortion measure amounts to calculating the energy of the ZSR vector $g\boldsymbol{H}\boldsymbol{c}_k$ and the correlation between this ZSR vector and the target vec-

tor $\boldsymbol{t}$. The adaptive codebook search can be formulated in the same way using the same kind of distortion measure.

Many efficient codebook search procedures have been proposed based on the distortion measure in (17.10). In addition, many special fixed codebook structures that allow efficient codebook search have been proposed in the literature. Together these techniques enabled the computational complexity to be reduced from the 430+ MFLOPS of the original CELP to 10 MFLOPS and below for many of the practical CELP coders deployed today.

17.7 US Federal Standard FS1016 CELP

The US Federal Standard FS1016 coder [17.20] is the first CELP coder ever standardized. It was developed by the US Department of Defense and is quite similar to the stochastically excited linear prediction (SELP) coder proposed in [17.36]. This coder uses a frame size of 30 ms. Each frame is divided into four subframes of 7.5 ms each (60 samples at 8 kHz sampling). The adaptive codebook approach described in [17.36] is used to determine the pitch period (also called the *pitch lag*) and the pitch gain once every subframe. (In this case, there is one excitation vector in each subframe.) For odd-numbered subframes, the pitch lag takes an integer value between 20 and 147 and thus can be encoded into 7 bits. For even-numbered subframes, the pitch lag is constrained to be within 32 samples relative to the pitch lag of the previous subframe. The pitch gain is coded using a five-bit nonuniform scalar quantizer. For the fixed codebook, the FS1016 standard uses a specially structured codebook where adjacent codevectors are essentially shifted version of each other and differ by only two samples at the end. This kind of shifted fixed codebook was first proposed in [17.39]. However, it was also independently studied in [17.36] using an improved distortion measure, and the optimal tradeoff of the two-sample shift was evaluated and proposed in [17.36].

This shifted fixed codebook introduces a certain constraint and structure into the codebook and allows an efficient codebook search. This is because when adjacent fixed codebook vectors are shifted version of each other, a large part of the ZSR computations in the matrix-vector multiplication $\boldsymbol{Hc}_k$ can be reused when going from one codevector to the next. Therefore, once the ZSR is calculated for the first codevector, the ZSR of the remaining codevectors can be obtained recursively with low complexity. This same principle also applies to the adaptive codebook search for those pitch lags that are not smaller than the excitation vector dimension (subframe size), since the corresponding adjacent adaptive codevectors are simply 1-sample shifted versions of each other.

To illustrate how a shifted codebook can reduce the codebook search complexity, first consider a fixed codebook with a one-sample shift between adjacent codevectors. In [17.39], a 1024-sample circular buffer $v(n), n = 0, 1, \ldots, 1023$ is used to store the 1024 codevectors of a 10-bit fixed codebook with 1-sample shift. The j-th codevector is defined to be $c_k(n) = v(k+n), 0 \le n \le N-1, 0 \le k \le 1023$, where $v(k+1024) = v(k)$ since $v(n)$ is in a circular buffer. Let $\boldsymbol{Hc}_k = \boldsymbol{r}_k = [r_k(0), r_k(1), \ldots, r_k(N-1)]^{\mathrm{T}}$. Then, due to the lower triangular Toeplitz structure of the $\boldsymbol{H}$ matrix, it is shown in [17.39] that $r_{k+1}(n)$ can be computed recursively from $r_k(n)$ as follows.

$$r_{k+1}(n-1) = r_k(n) - v(k)h(n)\,, \quad 1 \le n \le N-1\,,$$

$$r_{k+1}(N-1) = \sum_{n=0}^{N-1} v(k+1+n)h(N-1-n)\,.$$

This recursion is best understood with a graphical interpretation of the matrix-vector multiplication $\boldsymbol{Hc}_k$. In such a multiplication, the column vector $\boldsymbol{c}_k$ is turned sideways by 90° counterclockwise, then each of its elements is sample-by-sample multiplied with each element of the corresponding column of the matrix $\boldsymbol{H}$. Next, all the product terms along each row of the matrix is summed. The resulting column vector is $\boldsymbol{r}_k$. In the recursion above proposed in [17.39], when going from $\boldsymbol{c}_k$ to $\boldsymbol{c}_{k+1}$, the top element of $\boldsymbol{c}_k$ is removed, the remaining $N-1$ element is shifted up by one sample, and a new element is added at the bottom. Due to the lower triangular Toeplitz structure of the matrix $\boldsymbol{H}$, most of the product terms and the partial sums needed for $\boldsymbol{r}_{k+1}$ are already calculated for $\boldsymbol{r}_k$ and thus can be reused. However, the contribution to $r_k(n)$ due to the removed top element of $\boldsymbol{c}_k$ is already added to the element of $\boldsymbol{r}_k$. That is why in the first equation of the recursion above, the product term contribution due to the top element of $\boldsymbol{r}_k$ needs to be subtracted out again. None of the product terms in the second equation of the recursion above involving $r_{k+1}(N-1)$ has been calculated when calculating $\boldsymbol{r}_k$, so the entire summation in this second

equation needs to be calculated. This recursion above costs $(N-1)+N=2N-1$ multiply–add operations for each new $\boldsymbol{r}_{k+1}$ vector.

In fact, there is an even more-efficient recursion that requires only $N-1$ multiply–add operations for each new $\boldsymbol{r}_k$ vector. The trick is to reverse the order of the $\boldsymbol{r}_k$ vectors in the recursion. Using the graphical interpretation of the matrix-vector multiplication $\boldsymbol{Hc}_k$ explained above, one can see that if the recursion now starts at $\boldsymbol{r}_{1023}$ and going backward toward $\boldsymbol{r}_0$, then each time when going from $\boldsymbol{c}_{k+1}$ to $\boldsymbol{c}_k$, the bottom (last) element of $\boldsymbol{c}_{k+1}$ is removed, the remaining elements are shifted down by one sample, and a new element is added at the top. However, for the first $N-1$ element of $\boldsymbol{c}_{k+1}$ that are being shifted down, their partial sum contributions to $\boldsymbol{r}_k$ are exactly the first $N-1$ elements of $\boldsymbol{r}_{k+1}$. After these first $N-1$ elements of $\boldsymbol{c}_{k+1}$ have been shifted down and a new element added at the top to get $\boldsymbol{c}_k$, their partial sum contributions to $\boldsymbol{r}_k$ correspond to the product terms in the second through the N-th columns of the matrix $\boldsymbol{H}$. Thus, only the product terms in the first column need to be added to the partial sums already calculated and stored in the first $N-1$ elements of $\boldsymbol{r}_{k+1}$. Thus, the recursion can be described as

$$r_k(n) = r_{k+1}(n-1) + v(k)h(n)\,, \quad 1 \le n \le N-1\,,$$
$$r_k(0) = v(k)h(0) = v(k)\,.$$

The last equality above holds because $h(0)=1$. Therefore, this recursion performed in the reverse order requires only $N-1$ multiply-add operations for each new $\boldsymbol{r}_k$ vector. As mentioned above, this recursion can be used for the adaptive codebook search for those pitch lags not smaller than the subframe size.

The effect of shifting more than one sample between adjacent codevectors in the fixed codebook has been studied and reported in [17.36]. As the amount of shift between adjacent fixed codebook vectors increases, the coder performance increases, but the codebook search complexity also increases. It was found in [17.36] that for a large codebook (with 128 or more codevectors), a two-sample shift gives the same performance as a fully independent codebook. For this reason, the FS1016 coder uses such a fixed codebook with a two-sample shift between adjacent codevectors.

The same basic idea in the more efficient recursion above can easily be extended to the fixed codebook with a two-sample shift. In this case, each time when going from $\boldsymbol{c}_{k+1}$ to $\boldsymbol{c}_k$, the bottom two elements of $\boldsymbol{c}_{k+1}$ are removed, the remaining elements are shifted down by two samples, and two new elements are added at the top. For the first $N-2$ element of $\boldsymbol{c}_{k+1}$ that are being shifted down, their partial sum contributions to $\boldsymbol{r}_k$ are exactly the first $N-2$ elements of $\boldsymbol{r}_{k+1}$. These partial sum contributions to $\boldsymbol{r}_k$ correspond to the product terms in the third through the N-th columns of the matrix $\boldsymbol{H}$. Thus, only the product terms in the first two columns need to be added to the partial sums already calculated and stored in the first $N-2$ elements of $\boldsymbol{r}_{k+1}$. Thus, this recursion takes only $(N-1)+(N-2)=2N-3$ multiply–add operations to calculate each new $\boldsymbol{r}_k$.

In the case of the FS1016 coder, the elements of the fixed codebook are obtained by using a sequence of zero-mean, unit-variance, white Gaussian random numbers that are center-clipped at 1.2, which results in roughly 75% of the samples being zero. Thus, the fixed codebook is not only shifted, but also 75% *sparse*.

A sparse fixed codebook is said to produce slightly improved perceptual quality of the CELP output speech [17.36,38]. Furthermore, a sparse fixed codebook also reduces the codebook search complexity [17.38]. The reason is quite simple. Consider the graphical interpretation of the matrix–vector multiplication $\boldsymbol{Hc}_k$ again. When 75% to 90% of the elements in $\boldsymbol{c}_k$ are zero, the product terms in the corresponding columns of the matrix $\boldsymbol{H}$ do not need to be calculated, and thus the corresponding calculations can be saved. According to [17.20], the shifted sparse fixed codebook structure reduces the computation by a factor of 20.

In the FS1016 coder, the fixed codebook gain is quantized to 5 bits using a non-uniform scalar quantizer. An interoperable coder may use only a subset of the fixed codebook to reduce the computational complexity. The FS1016 coder also uses unequal forward error correction to protect perceptually most sensitive bits, and it uses extensive parameter smoothers to reduce the quality degrading effects of bit errors. At the time it was standardized, the output speech quality of the FS1016 coder was considered one of the best at around 4.8 kb/s.

17.8 Vector Sum Excited Linear Prediction (VSELP)

Vector sum excited linear prediction (VSELP) [17.17, 40] is another class of CELP coder that uses a specially structured fixed codebook to reduce the codebook search complexity. VSELP is the second type of CELP coder that was standardized.

Each codevector in the fixed codebook of VSELP is constructed as a weighted sum of M independent basis vectors, with the weights taking only two possible values: $+1$ or -1. Thus, an M-bit fixed codebook uses exactly M basis vectors. Let $\boldsymbol{v}_m = [v_m(0), v_m(1), \ldots, v_m(N-1)]^{\mathrm{T}}, m = 1, 2, \ldots, M$ be the M basis vectors. Then, the 2^M codevectors in an M-bit VSELP fixed codebook is constructed as

$$\boldsymbol{c}_k = \sum_{m=1}^{M} \theta_{km} \boldsymbol{v}_m \tag{17.11}$$

for $k = 0, 1, \ldots, 2^M$, where $\theta_{km} = +1$ if bit m of the k-th codeword is 1, and $\theta_{km} = -1$ if bit m of the k-th codeword is 0.

An 8 kb/s VSELP coder [17.17] was selected as the TIA interim standard IS-54 for North American digital cellular telephone applications, although this IS-54 standard was later rescinded. A 6.7 kb/s VSELP coder was selected as the Japanese digital cellular PDC full-rate (PDC-FR) standard coder, and a 5.6 kb/s VSELP coder [17.21] also became the GSM half-rate (GSM-HR) standard coder.

In the IS-54 VSELP coder, there are two fixed codebooks sequentially searched, similar to a multistage VQ configuration. In the PDC-FR VSELP coder, only one fixed codebook is used. In the GSM-HR VSELP coder, there are two fixed codebooks, but how many fixed codebooks are used depends on the voicing mode. For example, for a voiced mode the adaptive codebook and one of the fixed codebooks is used, but for an unvoiced mode the adaptive codebook is not used and two fixed codebooks are used.

In the IS-54 VSELP coder, the adaptive codebook and the two fixed codebooks are treated like three successive stages of quantization. The codebook gains are left *floating* when searching for the codebook vectors of these three stages. The mathematics of the efficient codebook search is somewhat involved. Due to the space limitation, only the basic concepts will be described below. Interested readers are referred to [17.17] for more mathematical details.

The adaptive codebook is first searched using the method described in Sect. 17.7. Next, when searching each of the two fixed codebooks, each of the M basis vectors are first filtered through the weighted short-term synthesis filter with zero initial memory to get the ZSR vector of that basis vector. Then, each of such ZSR vectors are orthogonalized with respect to each other and with respect to the ZSR vector due to the selected adaptive codebook vector. The ZSR vector due to each of the 2^M fixed codebook vectors can then be expressed as a linear combination of the M orthogonalized ZSR vectors due to the M basis vectors.

The search procedure needs to find the fixed codebook vector that minimizes a distortion measure that is the ratio of two quantities:

1. the square of the correlation between the target vector and the ZSR vector due to the fixed codebook vector
2. the energy of the ZSR vector due to the fixed codebook vector

These two quantities can each be evaluated in a recursive manner with low computational complexity if the codebook search procedure sequences through the codevectors using a binary Gray code so that the codewords of the adjacent codevectors differ by only one bit.

Another factor of two saving can be obtained by observing that the two fixed codevectors corresponding to the two codewords that are 1s complement of each other have the same shape but only differ in sign. Thus, the two quantities above for the distortion measure will be the same for these two complementary codevectors. The winner of the two complementary codevectors can be determined easily by just examining the sign of the correlation term in the first quantity. This means that only half as many distortion values need to be calculated.

Another novel feature of the IS-54 VSELP coder is the way it quantizes the gains of the adaptive codebook and the fixed codebooks. First the adaptive codebook vector and fixed codebook vectors are identified by the codebook search procedure outlined above. After that, rather than separate scalar quantization of each of the codebook gains as was the normal procedure in previous coders, this VSELP coder first encodes the speech energy of the entire 20 ms frame. Then, the approximate excitation signal energy in each of the 5 ms subframes is calculated based on the frame energy of speech and the reflection coefficients in each subframe. Next, the adaptive codebook gain and the two fixed codebook gains in each subframe are converted to the energy domain expressed as fractions of the total excitation energy in that subframe. The resulting three energy fractions are then jointly vector quantized using a trained codebook. At the VSELP decoder, the three decoded energy fractions are converted back to the codebook gain domain.

It is said [17.17] that such a gain VQ scheme in the energy-fraction domain not only makes the resulting energy fractions more correlated which allows more efficient VQ coding, but also the fact that all codebook gains are represented as energy fractions makes the en-

ergy of the decoded speech less susceptible to bit errors in the codebook gain codeword.

The GSM-HR VSELP coder also has another feature not yet discussed in this chapter – a long-term predictor with a fractional (non-integer) pitch period [17.41, 42]. Rather than using an integer pitch period as in previous predictive coders, an interpolation filter is used to achieve the effect of a non-integer pitch period. This allows the long-term synthesis filter to model the pitch periodicity of voiced speech with a higher temporal resolution, thus resulting in higher output speech quality.

In VSELP, the basis vectors for the fixed codebook can be optimized using a training database. It is reported [17.17] that such optimization of the basis vectors resulted in 0.64 dB improvement in weighted segmental SNR and significant improvement in subjective quality.

17.9 Low-Delay CELP (LD-CELP)

The low-delay CELP (LD-CELP) coder [17.5, 6, 43] is another different kind of CELP coder. It is the third kind of CELP coder that was selected as a standard coder. The ITU-T recommendation G.728 standard coder [17.6] is based on it. There are several important differences between G.728 and the previous CELP coders. These differences will be described in this section.

First of all, the excitation vector dimension used in G.728 LD-CELP is much shorter than that of earlier CELP coders. Most of the earlier CELP coders used an excitation vector dimension of 40 samples (5 ms) or more in order to get a better excitation VQ efficiency (since VQ performance generally increases with increasing vector dimension). However, using such a large vector dimension will not meet the ITU-T requirement of a very low coding delay. In order to achieve a one-way coding delay of less than 2 ms (16 samples at 8 kHz sampling), the G.728 coder is forced to use a frame size and vector dimension of merely five samples (0.625 ms).

With a frame size of five samples, basically the G.728 encoder is required to produce 10 bits as output for every five samples of input speech. This strict constraint, which is due to the low-delay requirement, greatly limits the flexibility of the coder design. While previous CELP coders could spend 20–40 bits per frame to encode the short-term predictor parameters and another 10 to 12 bits per subframe to encode the long-term predictor parameters, the G.728 coder cannot do any of these since it only has 10 bits total to encode everything in a frame.

This leads to the second major difference between G.728 and previous CELP coders – the predictor parameters are made *backward-adaptive*. Previous CELP coders all used *forward-adaptive* predictors, where the optimal predictor parameters are derived from the input speech and then quantized and transmitted to the decoder. With backward adaptation, no bits are spent in sending the parameters to the decoder; instead, the parameters are locally derived at both the encoder and the decoder from previously decoded speech signal or parameters. See [17.13] for a more-detailed discussion of backward adaptation versus forward adaptation. For G.728, both the short-term synthesis filter coefficients and the excitation gain are backward-adaptive and thus does not require any bit to transmit them to the decoder.

The third major difference between G.728 and previous CELP coders is the elimination of the long-term predictor, and in its place, the use of a high-order backward-adaptive short-term predictor. The reason is that a backward-adaptive long-term predictor has a strong tendency to diverge at the decoder due to bit errors or other reasons that cause a mismatch of the encoder states and the decoder states. On the other hand, when the backward-adaptive short-term predictor coefficients are obtained by performing LPC analysis on previously decoded speech signal, it is relatively easy to maintain the convergence at the decoder even with bit errors and states mismatch. It was found [17.5] that, with the short-term predictor order set at the conventional value of 10, the lack of a long-term predictor degrades female speech quality significantly. On the other hand, if the short-term predictor order is allowed to increase to 50, then a 50-th-order short-term predictor can exploit the pitch periodicity in female speech, since most female voices have a pitch period of less than 50 samples.

Using a 50-th-order short-term predictor in a conventional forward-adaptive CELP coder would be impractical due to the large number of predictor coefficients that need to be transmitted. However, since G.728 uses backward adaptation to update the short-term predictor, it doesn't cost any bit to increase the predictor order to 50. The only price paid is the increased computational complexity to derive the 50 predictor coefficients. In order to achieve toll quality

at 16 kb/s and with a frame size of only five samples, it was necessary to exploit the pitch periodicity in speech by using a 50-th-order short-term predictor. Thus, under the severe low-delay constraint, the G.728 had to make the complexity-quality trade-off – achieving toll quality through the use of a higher-complexity 50-th-order backward-adaptive short-term predictor.

The encoder structure of the G.728 LD-CELP coder is equivalent to the structure shown in Fig. 17.3 but without the long-term synthesis filter and the signal modifier. Both the excitation gain and the short-term synthesis filter are backward-adaptive. The excitation shape codebook (the fixed codebook) is a 10-bit codebook for five-dimensional VQ. However, to control the computational complexity of the codebook search, a gain-shape structured codebook is used, with the codebook constructed as the product of a 3-bit gain codebook and a 7-bit shape codebook. The three gain bits consists of one sign bit and two magnitude bits. The 7-bit shape codebook contains 128 independent codevectors.

Both the 7-bit shape codebook and the 2-bit magnitude codebook are closed-loop-trained based on the weighted distortion measure of LD-CELP using a large training speech file. Thus, the effects of backward adaptation of the predictor and gain are automatically taken into account in the training. Such closed-loop codebook training gives significant audio quality improvement and is crucial for achieving good speech quality in LD-CELP.

17.10 Pitch Synchronous Innovation CELP (PSI-CELP)

Pitch synchronous innovation CELP (PSI-CELP) [17.44] is the fourth kind of CELP coder that was selected as a standard coder. For the personal digital cellular (PDC) system of Japan, the half-rate standard coder (PDC-HR) is a 3.45 kb/s PSI-CELP coder [17.22].

The main distinguishing feature of PSI-CELP is that the fixed codebook is made adaptive by repeating the first portion of each codevector in a pitch-synchronous manner if the pitch period is smaller than the excitation vector dimension. Specifically, if the pitch period T is less than the vector dimension N, then the first T samples of the fixed codebook vectors are periodically repeated for later samples starting at the $(T+1)$-th sample. This is important for the PDC-HR coder because the coder uses a fairly large subframe size (vector dimension) of 80 samples (10 ms), which is larger than many typical pitch period values.

In place of the typical adaptive codebook in other CELP coders, the PDC-HR standard PSI-CELP coder actually uses either an adaptive codebook or a fixed codebook, with the fixed codebook mainly used in non-periodic regions of the speech signal. The adaptive codebook uses fractional pitch period [17.41, 42]. To save the codebook storage requirement, the fixed codebook has four basis vectors each rotated eight times to get 32 codevectors.

In place of the typical fixed codebook in other CELP coders, the PDC-HR standard PSI-CELP coder uses two stochastic codebooks in a conjugate structure [17.45] (which is similar to two-stage VQ). Each of the two stochastic codebooks contains 16 vectors. The selected codevector from each codebook is multiplied by a sign and the resulting two codevectors are added together. The pitch synchronous innovation (PSI) procedure is applied to the stochastic codebooks. To reduce the codebook search complexity, six out of the 16 codevectors from each stochastic codebook are preselected using a simplified method. Only the preselected codevectors go through full-complexity search.

The adaptive/fixed codebook gain and the stochastic codebook gain are jointly vector quantized to seven bits per subframe in a way somewhat similar to the codebook gain quantization of VSELP [17.17]. A delayed-decision coding technique is used to improve the PSI-CELP performance. Two best candidate codevectors are selected from the adaptive/fixed codebook search, and the remaining quantization procedures are performed for each candidate. The candidate that gives the lower distortion in the fully quantized version is selected as the winner.

17.11 Algebraic CELP (ACELP)

Algebraic code-excited linear prediction (ACELP) signifies the use of algebraic codes to make up the excitation in CELP. Due to the popularity of ITU-T recommendation G.729 [17.46] many people think of ACELP [17.24]

and G.729 as synonymous. However, in reality it is only the excitation of G.729 that is algebraic. Furthermore, G.729 and other speech coding standards utilizes sparse algebraic codes, (SACs) [17.24], and not an algebraic code as originally proposed in [17.23] in 1987 for CELP. The sparseness contributes significantly to a low computational complexity. Of further interest, a number of the techniques and structural changes proposed in [17.23] to the original CELP structure [17.3, 4] form the cornerstones of G.729 and many other speech coding standards. It should be noted that partly overlapping techniques were proposed in [17.38] in 1986. These techniques and structures are reviewed in other relevant sections, e.g., Sect. 17.6.

As indicated above, although the abbreviation ACELP was introduced in 1990 in [17.24], earlier work on algebraic codes for CELP appeared in 1987 in [17.23] and [17.47]. A significant advancement is the notion of sparse algebraic codes [17.24]. The idea of using sparse excitation in CELP coding was introduced in [17.38] in 1986, but in the context of Gaussian excitation vectors and not algebraic codes. Hence, the pulse amplitudes would be Gaussian distributed and not binary $(+1, -1)$ as in sparse algebraic codes. In [17.38], the use of sparse excitation for CELP is compared to MPLPC (multipulse linear predictive coding) [17.2] in terms of the number of nonzero pulses for voiced speech. Other early publications on sparse pulse excitation for CELP include [17.48] and [17.49].

ACELP has been a dominating form of excitation as the fixed codebook excitation in CELP speech coding standards from the mid 1990s until today (2006).

17.11.1 ACELP Background

ACELP appears to originate from the idea of using binary error correcting codes to represent N points on an M dimensional hyper sphere, i. e., a fixed codebook of size N and vector dimension M [17.47]. Even earlier, [17.50] proposed a spherical vector quantizer for encoding of the 1000 Hz base band of the short-term prediction residual after eighth-order LPC. The justification for the view of the fixed codebook representing points on a hyper sphere originates from the separate gain of the fixed codebook. Basically, the length of the residual vector, that is being approximated by the fixed codebook, can be considered normalized. Hence, the residual vectors will be points on a hypersphere. The task of the fixed codebook is to populate the hyper sphere in an optimal way. If the common assumption, that the samples of the residual vector after short- and long-term prediction are independent and identically distributed (i.i.d.) Gaussian, is reasonable, then the normalized residual vectors in terms of points on the hypersphere will be a uniform distribution. All $(+1, -1)M$-tuples will represent a uniform discrete sampling of the hypersphere with 2^M points, and it seems reasonable to pick the fixed codebook as a subset of the $2^M(+1, -1)M$-tuples. Error-correcting codes are attractive for picking the subset as they basically maximize the minimum distance between any two codewords, codevectors in the context of a fixed codebook, and spread out codewords for maximum coverage. Naturally, a mapping from the $(0, 1)M$ bits of a error correcting codeword to the $(+1, -1)M$ elements of the codevector of the fixed codebook is required. The benefit of such a fixed codebook is that no storage is required and it lends itself well to efficient methods.

An important additional feature is the realization that only relatively few nonzero pulses are required. For analysis-by-synthesis multipulse with only short-term prediction, the need for few nonzero pulses was reported in [17.2], where little improvement was observed after 8 pulses per 80 samples (10 ms). Similarly in the context of CELP, i. e., with long-term prediction as well, [17.38] reported the need for few nonzero elements. Although [17.38] includes long-term prediction an equivalent of four pulses per 40 samples (5 ms) was reported. However, neither [17.2] nor [17.38] were considering pulses of binary amplitude, but instead pulses of arbitrary amplitude. In [17.48,51,52], and [17.49] sparse binary pulse codevectors were discussed, and in [17.24] it was proposed along with the definition of the abbreviation ACELP and SAC. Technically, sparse $(+1, -1)$ binary excitation is really $(+1, 0, -1)$ ternary excitation.

The concept of pulse tracks and interleaved permutation codes [17.53, 54] contribute to further reducing complexity of searching and bit-rate of coding the sparse binary pulses. Partitioning the sample of an excitation vector into multiple tracks and applying a permutation code (resulting in sparse binary pulses) to each track has proven effective. Furthermore, interleaving the tracks is typical as it allows the flexibility of high local density of pulses in the final fixed codebook excitation.

Table 17.1 Example ACELP ISPP excitation structure

Track/code	Sample positions	Bits
1	0, 5, 10, 15, 20, 25, 30, 35	3 + 1 = 4
2	1, 6, 11, 16, 21, 26, 31, 36	3 + 1 = 4
3	2, 7, 12, 17, 22, 27, 32, 37	3 + 1 = 4
4	3, 8, 13, 18, 23, 28, 33, 38	3 + 1 = 4
5	4, 9, 14, 19, 24, 29, 34, 39	3 + 1 = 4

Table 17.1 shows an example of a 40 sample ACELP fixed codebook made up of five single pulse interleaved permutation codes (5 tracks). Hence, it is a so-called interleaved single pulse permutation (ISPP) design. It results in a $5 \cdot (3+1) = 20$-bit fixed codebook excitation with five pulses. Other ACELP interleaved permutation codes employ multiple pulses per code/track [17.55].

Key advantages of ACELP are that it

- lends itself well to efficient methods,
- eliminates the need to store codebook,
- offers good speech quality,
- is flexible, and
- enables the use of large codebooks.

17.11.2 ACELP Efficient Search Methods

Modern ACELP is typically used in conjunction with the adaptive codebook implementation of the long-term (pitch) synthesis filter. Hence, the present section will present the search methods of ACELP in that context. Furthermore, the perceptual weighting filter of modern ACELP coders is frequently given by

$$W(z) = \frac{A(z/\gamma_1)}{A(z/\gamma_2)} , \tag{17.12}$$

where $A(z)$ is the unquantized prediction error filter and $0 < \gamma_2 < \gamma_1 < 1$. This form is used here without limitation, and all equations are easily extended to accommodate any perceptual weighting filter.

The efficient search methods for ACELP are based on a target signal, $t(n)$, which is basically the weighted input speech from which contributions from

1. ringing of filter memory
2. adaptive codebook

have been subtracted. Hence, the target signal, $t(n)$, is in the weighted speech domain. Searching the ACELP fixed codebook involves finding the entry that minimizes the MSE between the target signal and the ACELP codevector passed through the perceptual weighting filter and short-term synthesis filter, both with zero memory. This is illustrated in Fig. 17.8, and it is expressed as

$$I = \arg\min_k (E_k) = \arg\min_k \left\{ \sum_{n=0}^{N-1} [t(n) - y_k(n)]^2 \right\} , \tag{17.13}$$

where $y_k(n)$ is the output from passing the k-th ACELP codevector through the perceptual weighting filter and short-term synthesis filter. In terms of the ACELP codevector $c_k(n)$ (17.13) is expressed as

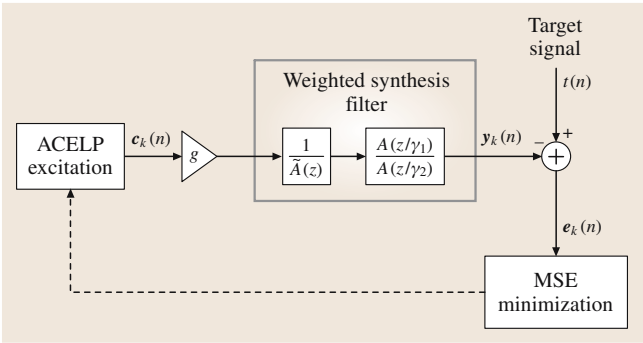


Fig. 17.8 Basic ACELP fixed codebook excitation search

$$I = \arg\min_k \left[\sum_{n=0}^{N-1} (t(n) - \{h(n) * [g \cdot c_k(n)]\})^2 \right] , \tag{17.14}$$

where $h(n)$ is the impulse response of the weighted synthesis filter, i. e., the inverse z-transform of

$$H(z) = \frac{1}{\tilde{A}(z)} W(z) = \frac{1}{\tilde{A}(z)} \frac{A(z/\gamma_1)}{A(z/\gamma_2)} , \tag{17.15}$$

where $1/\tilde{A}(z)$ is the quantized short-term synthesis filter. Note that enhancements to the ACELP codebook can be included into the impulse response, $h(n)$, without affecting the following search techniques. The enhancements are in the form of a prefilter, $F(z)$. One such example is the in-frame pitch enhancement for pitch lags shorter than the length of the ACELP codevector [17.25, 56]:

$$F(z) = \frac{1}{1 - \beta Z^{-T}} , \tag{17.16}$$

where T is the pitch lag, and β is a suitable, typically adaptive, filter coefficient related to the periodicity. This particular ACELP pitch prefilter is incorporated into $h(n)$ according to

$$\begin{aligned} h(n) &\leftarrow h(n) + \beta h(n-T) , \\ n &= T, T+1, \ldots , N-1 . \end{aligned} \tag{17.17}$$

In AMR-WB (adaptive multi-rate) [17.57], the ACELP prefilter additionally includes a tilt part:

$$F(z) = \frac{1}{1 - \beta Z^{-T}} (1 - \alpha z^{-1}) , \tag{17.18}$$

where also α is adaptive.

Equation (17.14) is written in vector form as

$$
\begin{aligned}
I &= \arg\min_k \left[(\boldsymbol{t} - g\boldsymbol{H}\boldsymbol{c}_k)^{\mathrm{T}} (\boldsymbol{t} - g\boldsymbol{H}\boldsymbol{c}_k) \right] \\
&= \arg\min_k \left[\boldsymbol{t}^{\mathrm{T}}\boldsymbol{t} + g^2 (\boldsymbol{H}\boldsymbol{c}_k)^T (\boldsymbol{H}\boldsymbol{c}_k) \right. \\
&\quad \left. - g\boldsymbol{t}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{c}_k - g (\boldsymbol{H}\boldsymbol{c}_k)^{\mathrm{T}} \boldsymbol{t} \right] \\
&= \arg\min_k \left(\boldsymbol{t}^{\mathrm{T}}\boldsymbol{t} + g^2 \boldsymbol{c}_k^{\mathrm{T}} \boldsymbol{H}^{\mathrm{T}} \boldsymbol{H}\boldsymbol{c}_k - 2g\boldsymbol{t}^{\mathrm{T}} \boldsymbol{H}\boldsymbol{c}_k \right) ,
\end{aligned}
\tag{17.19}
$$

where

$$
\begin{aligned}
\boldsymbol{t} &= [t(0)t(1)\cdots t(N-1)]^{\mathrm{T}} , \\
\boldsymbol{c}_k &= [c_k(0)c_k(1)\cdots c_k(N-1)]^{\mathrm{T}} , \\
\boldsymbol{H} &= \begin{pmatrix}
h(0) & 0 & \cdots & 0 \\
h(1) & h(0) & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
h(N-2) & h(N-3) & \cdots & 0 \\
h(N-1) & h(N-2) & \cdots & h(0)
\end{pmatrix} .
\end{aligned}
$$

According to common procedure in ACELP, the excitation, $\boldsymbol{c}_k$, is found under the assumption of optimal gain g.

Table 17.2 Key properties of ACELP excitation in standards

Standard	Rate (kbps)	ACELP frame size (samples)	Pulses	Tracks	Bits
G.723.1	5.3	60	4	4	17
G.729		40	4	4	17
G.722.2/AMR-WB	23.85 / 23.05	64	24	4	88
	19.85		18	4	72
	18.25		16	4	64
	15.85		12	4	52
	14.25		10	4	44
	12.65		8	4	36
	8.85		4	4	20
	6.6		2	2	12
GSM AMR	12.2 (GSM EFR)	40	10	5	35
	10.2		8	4	31
	7.95 / 7.4 (TIA EFR)		4	4	17
	6.7 (PDC EFR)		3	3	14
	5.9		2	2	11
	5.15 / 4.75		2	5	9
EVRC	Full-rate	53 / 54	8	5	35
	Half-rate		3	3	10
VMR-WB	Full-rate	64	8	4	36
	Voiced and generic half-rate		2	2	12
MPEG-4	8 kHz core		3-12		
	8 kHz enhancement	40	2		
SMV	Rate 1/1 type 1	40	8	8	30
	Rate 1/1 type 0	40	5	5	21
			5	5	20 ⇒ 22
			5	4	20
	Rate 1/2 type 1	53/54	2	2	12
			3	3	11 ⇒ 13
			5	5	11
	Rate 1/2 type 0	80	2	2	14
			3	3	13 ⇒ 15
			Gaussian	NA	13

The gain is determined by minimizing the error energy:

$$\frac{\partial E_k}{\partial g} = \frac{\partial}{\partial g}\left(\boldsymbol{t}^{\mathrm{T}}\boldsymbol{t} + g^2\boldsymbol{c}_k^{\mathrm{T}}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{c}_k - 2g\boldsymbol{t}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{c}_k\right) = 0$$

$$\Downarrow$$

$$g = \frac{\boldsymbol{t}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{c}_k}{\boldsymbol{c}_k^{\mathrm{T}}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{c}_k} \,. \tag{17.20}$$

Inserting this result in (17.19) yields

$$I = \arg\min_k \left(\boldsymbol{t}^{\mathrm{T}}\boldsymbol{t} - \frac{\left(\boldsymbol{t}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{c}_k\right)^2}{\boldsymbol{c}_k^{\mathrm{T}}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{c}_k}\right) \,. \tag{17.21}$$

Since $\boldsymbol{t}$ is independent of the codevector, $\boldsymbol{c}_k$, this is equivalent to

$$I = \arg\max_k \left(\frac{\left(\boldsymbol{t}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{c}_k\right)^2}{\boldsymbol{c}_k^{\mathrm{T}}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{c}_k}\right) \,. \tag{17.22}$$

The numerator is often rewritten in terms of the backward filtered target vector [17.23,53], $\boldsymbol{t}_{\mathrm{b}} = \boldsymbol{H}^{\mathrm{T}}\boldsymbol{t}$, and the search is expressed as

$$I = \arg\max_k \left(\frac{\left(\boldsymbol{t}_{\mathrm{b}}^{\mathrm{T}}\boldsymbol{c}_k\right)^2}{\boldsymbol{c}_k^{\mathrm{T}}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{c}_k}\right) = \arg\max_k \left(\frac{\left(\boldsymbol{t}_{\mathrm{b}}^{\mathrm{T}}\boldsymbol{c}_k\right)^2}{\boldsymbol{c}_k^{\mathrm{T}}\boldsymbol{\Phi}\boldsymbol{c}_k}\right) \,, \tag{17.23}$$

where $\boldsymbol{\Phi} = \boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}$ is symmetric and contains the autocorrelation of the impulse response of $H(z)$:

$$\Phi(i,\,j) = \begin{cases} \sum_{l=0}^{N-1-i} h(l)h(l+i-j) \,, & j \le i \\ \Phi(j,i) \,, & j > i \end{cases} \,. \tag{17.24}$$

The backward filtered target $\boldsymbol{t}_{\mathrm{b}}$ and the autocorrelation matrix $\boldsymbol{\Phi}$ are calculated prior to searching the ACELP codebook.

The efficient search methods all explore ways to most efficiently evaluate (17.23), either in a mathematically equivalent way, or in a slightly suboptimal way with only minor sacrifice in performance. The following subsections will present some of the methods typically used in ACELP.

Sparse ACELP

In sparse ACELP only relatively few pulses are nonzero, and if they are binary, the nonzero pulses take on an amplitude of ± 1. The P pulses, $P < N$, are described uniquely by the location and amplitude, m_i and a_i, respectively, $i = 0, 1, \ldots, P-1$. Note that $a_i = \pm 1$. With this notation, the search of (17.23) can be written as [17.53]

$$I = \arg\max_{\boldsymbol{m}\times\boldsymbol{a}} \left(\frac{\left(\sum_{i=0}^{P-1} t_{\mathrm{b}}(m_i)a_i\right)^2}{\sum_{i=0}^{P-1}\Phi(m_i,m_i) + 2\sum_{i=0}^{P-2}\sum_{j=i+1}^{P-1} a_i a_j \Phi(m_i,m_j)}\right) \,, \tag{17.25}$$

where the search is identifying the optimal P pulse positions $\boldsymbol{m} = (m_0, m_1, \ldots, m_{P-1})$ and P amplitudes $\boldsymbol{a} = (a_0, a_1, \ldots, a_{P-1})$. This search can be performed in a nested loop so that the addition of one more pulse is considered in each loop. At the inner most loop, the final cost function for a given candidate is calculated by one additional addition and the square to get the numerator, and P additional additions and one multiplication to get the numerator. This provides for a very efficient exhaustive search, but as the size of the codebook increases this quickly becomes impractical [17.53]. Sparse ACELP is used in all ACELP standards.

A Priori Sign

Studying the numerator of (17.25) it is evident that the cross correlation will be maximized by setting the pulse amplitudes, $\boldsymbol{a}$, basically the pulse signs, to be the same as the sign of the backward filtered target, $\boldsymbol{t}_{\mathrm{b}}$ [17.58]. Accordingly, by setting the pulse signs a priori only the pulse locations $\boldsymbol{m}$ need to searched. This is achieved by incorporating the a priori pulse signs into the backward filtering target and correlation matrix prior to the search [17.58] according to:

$$\tilde{t}_{\mathrm{b}}(n) = |t_{\mathrm{b}}(n)| \,, \tag{17.26}$$

$$\tilde{\Phi}(i,\,j) = \mathrm{sign}[t_{\mathrm{b}}(i)]\mathrm{sign}[t_{\mathrm{b}}(j)]\Phi(i,\,j) \,. \tag{17.27}$$

and performing the search for pulse positions according to

$$I = \arg\max_{\boldsymbol{m}} \left(\frac{\left(\sum_{i=0}^{P-1} \tilde{t}_{\mathrm{b}}(m_i)\right)^2}{\sum_{i=0}^{P-1}\tilde{\Phi}(m_i,m_i) + 2\sum_{i=0}^{P-2}\sum_{j=i+1}^{P-1}\tilde{\Phi}(m_i,m_j)}\right) \,, \tag{17.28}$$

where, again, the search identifies the optimal P pulse positions $\boldsymbol{m} = (m_0, m_1, \ldots, m_{P-1})$. To eliminate the multiplication by 2 during the search, all off-diagonal elements of $\tilde{\Phi}(i,\,j)$ can be multiplied by 2 prior to the search. The a priori determination of the pulse signs is

Table 17.3 Effectiveness of ACELP focused search [17.53]

Percentage of exhaustive search	SNR (dB)
100	22.20
4	22.14
1.6	22.00
0.2	22.05
0.15	21.83
0.05	21.80
0.03	21.50

not in general equivalent to the exhaustive search given by (17.25). It is used in G.729 [17.46], G.723.1 [17.25], EVRC [17.26], and GSM-EFR [17.56] among others. Note that the presetting of the sign can be based on the sign of a signal that is a linear combination of the backward filtered target and the long-term prediction residual [17.56] (or a similarly suitable signal), and hence not only based on the sign of the backward-filtered target.

Focused Search

Various focused search methods exist [17.53, 54, 58]. Generally, the P pulses are searched in nested loops and constraints on the correlation,

$$C = \sum_{i=0}^{p-2} \tilde{t}_b(m_i) , \quad p < P , \tag{17.29}$$

are set up to disregard subsections of the ACELP codebook (represented by entry into the p-th loop) if the correlation C does not meet a predefined threshold. A table from [17.53] is reproduced in Table 17.3, illustrating this in the form of the relation between the percentage of exhaustive search and the SNR for a sentence spoken by a female talker. The threshold can be calculated in various ways. It can be based on the maximum possible correlation of the existing $p-1$ pulse candidates, or the average correlation over their possible positions, or a combination thereof as in [17.58]. Generalizing [17.58] leads to a threshold based on maximum pulse correlation

$$C_{\max} = \sum_{l=0}^{p-2} \max_{i \in \mathcal{T}_l} \left[\tilde{t}_b(i)\right] \tag{17.30}$$

and average pulse correlation

$$C_{\text{avg}} = \sum_{l=0}^{p-2} \text{avg}_{i \in \mathcal{T}_l} \left[\tilde{t}_b(i)\right] . \tag{17.31}$$

In (17.30) and (17.31), $\mathcal{T}_l$ is the set of possible pulse positions of pulse l. The threshold can then be constructed as a combination of the maximum and average correlation:

$$C_{\text{thr}} = C_{\text{avg}} + \alpha_{\text{thr}}(C_{\max} - C_{\text{avg}}) . \tag{17.32}$$

This will reduce the average complexity. To have a firm limit on the worst-case complexity a threshold on the number of times a certain loop can be entered may be enforced [17.58]. A very large part of the ACELP codebook can be ruled out this way.

Depth-First Tree Search Procedure

As the number of pulses P increases, additional methods can be used to search the effectively increasing codebook efficiently. One such method is used in G.729 Annex A [17.59], GSM-EFR [17.56], and AMR-WB [17.57]. The depth-first tree search procedure would typically be used instead of the focused search to further reduce complexity. The set of P pulses is divided into N_m subsets, typically with an equal number of M pulses in each. While the subsets are searched sequentially, the pulses in each subset are searched jointly. First subset 1, then subset 2, etc. Subsequent sets are searched given the pulses of previous sets. This search is performed iteratively, circulating the assignment of pulses to tracks. For instance, at the first iteration the first two pulses can be assigned to tracks 0 and 1, while for the second iteration the first two pulses can be assigned to tracks 2 and 3, etc. In the second iteration the last two pulses can be assigned to tracks 0 and 1.

A reference signal may be constructed to assist in selecting a subset of pulse positions to consider in a given track. As an example, with pulses 0 and 1 being assigned to tracks 2 and 3 (each of eight positions), based on the reference signal, only four positions in track 2 are preselected for consideration for pulse 0. Such preselection reduces complexity. Techniques like this can be used to control the complexity and balance complexity of multiple rates in a speech coder. The AMR-WB speech coder [17.57] provides a good example of this technique. The reference signal is typically identical to the signal used to set the signs a priori (Sect. 17.11.2). It can be the backward filtered target signal as in [17.60], a linear combination of the backward filtered target and the long-term prediction residual signals as in [17.57], or some other suitable signal.

In G.729A, the depth-first tree search is used instead of the focused search of G.729 and provides a direct comparison. It results in a very significant saving of 5 MIPS (million instructions per second) at the cost of

a slight degradation in performance, about 0.2 dB SNR according to [17.60].

17.11.3 ACELP in Standards

A number of speech coding standards utilizes the ACELP excitation, that is, the sparse binary pulse excitation. In grouping by standards body: ITU-T G.723.1 [17.25] (ACELP for the 5.3 kbit/s rate, multipulse for the 6.3 kbit/s rate), ITU-T G.729 [17.46], ITU-T G.722.2 [17.61], ETSI GSM-EFR (European Telecommunications Standards Institute) [17.56], ETSI/3GPP GSM AMR (3-rd Generation Partnership Project) [17.62], 3GPP GSM AMR-WB [17.57], TIA-127 EVRC (enhanced variable rate codec) [17.26], TIA-136 EFR [17.63], TIA/3GPP2 SMV [17.64], 3GPP2 VMR-WB [17.28], ARIB PDC-EFR (Association of Radio Industries and Businesses), and MPEG-4 CELP (Moving Pictures Expert Group) narrow-band audio [17.65].

It should be noted that ITU-T G.722.2 and 3GPP GSM AMR WB is the same wide-band multirate speech coder. It was initially standardized for cellular by 3GPP and subsequently submitted to the ITU for consideration in an, at the time, ongoing standardization effort of wide-band coding of speech at around 16 kbit/s. It should also be noted that the AMR-WB rate of 12.65 kbit/s is interoperable with one of the rates of 3GPP2 VMR-WB. Furthermore, the GSM EFR is identical to the 12.2 kbit/s rate of ETSI/3GPP AMR, TIA-136 EFR is equivalent to the 7.4 kbit/s rate of AMR, and ARIB PDC-EFR is equivalent to the 6.7 kbit/s rate of AMR. Note that it can be somewhat confusing that multiple standards bodies use the same acronym, EFR. The MPEG-4 CELP is really somewhat of a hybrid between ACELP and multipulse as it doesn't use binary or ternary pulses, but instead apply a VQ of the pulse amplitudes. Similarly, TIA/3GPP2 SMV is a hybrid as it is based on sub-codebooks where most sub-codebooks are ACELP-like, but one is different, and the codebooks are weighted differently in selecting the overall best excitation.

Table 17.2 summarizes some of the key properties of the ACELP fixed codebook excitation structures used in the standards listed above.

17.12 Conjugate Structure CELP (CS-CELP) and CS-ACELP

Conjugate structure CELP (CS-CELP) introduces conjugate structure VQ for CELP. It was originally proposed in [17.66] for CELP. However, earlier the conjugate structure was used in other coders, e.g., in transform coding [17.67]. Generally, conjugate structure VQ constructs the output codevector $\boldsymbol{c}_{i,j}$ as a linear combination of the output codevectors from two codebooks $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$ respectively:

$$\boldsymbol{c}_{i,j} = \alpha_1 \cdot \boldsymbol{c}_i + \alpha_2 \cdot \boldsymbol{c}_j \,. \tag{17.33}$$

The advantages of the conjugate structure compared to a single VQ is threefold [17.66]:

- improves resilience to bit-errors,
- reduces memory requirement,
- facilitates reduced complexity methods.

These advantages are demonstrated in [17.66] and [17.68], which also present methods to train the conjugate structure codebooks. Furthermore, [17.66] and [17.68] present an 8 kb/s CS-CELP speech coder where the conjugate structure is used for both the fixed codebook excitation and the joint quantization of the subframe based adaptive codebook gain and fixed codebook gain. This coder was submitted by NTT for the ITU-T (CCITT at the time) G.729 standardization. Although the coder was not standardized as G.729, techniques from multiple candidates were merged and eventually formed G.729. The resulting G.729 standard [17.46] uses the conjugate structure for the 2-dimensional joint VQ of the subframe based adaptive codebook gain and fixed codebook gain. In G.729 the linear combination of the codevectors from the two codebooks is a simple summation, and the conjugate structure codevector is

$$\boldsymbol{c}_{i,j} = \boldsymbol{c}_i + \boldsymbol{c}_j \,. \tag{17.34}$$

One codebook has a bias towards the element corresponding to the fixed codebook gain, while the other codebook has a bias towards the element corresponding to the adaptive codebook gain. This allows open-loop preselection of both codebooks based on the respective dominant parameter. This leaves only a subset (a quarter) for the more-complex joint closed-loop search without any noticeable degradation compared to an exhaustive closed-loop search [17.58].

17.13 Relaxed CELP (RCELP) – Generalized Analysis by Synthesis

Relaxed CELP (RCELP) [17.69] has become synonymous with a specific and practical usage of the generalized analysis-by-synthesis principle proposed in [17.15,70]. The basic idea of generalized analysis-by-synthesis is to relax the waveform matching constraint of CELP without affecting the speech quality. In principle, a relaxation of the waveform matching can be incorporated into the error criterion. However, generalized analysis-by-synthesis proposes a general signal modification function, that is applied to the original signal, constrained to provide a perceptually similar signal. The idea is to modify the signal into a signal that is simpler to represent (in the sense of minimizing distortion relative to the modified signal), yet perceptually indistinguishable from the original. The fundamental and the generalized analysis-by-synthesis paradigms as presented in [17.15] are depicted in Figs. 17.9 and 17.10, respectively.

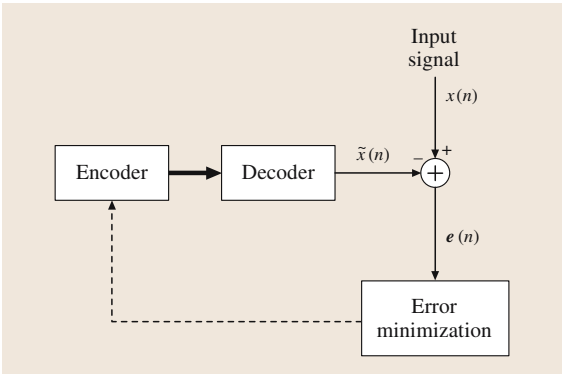


Fig. 17.9 Basic analysis by synthesis

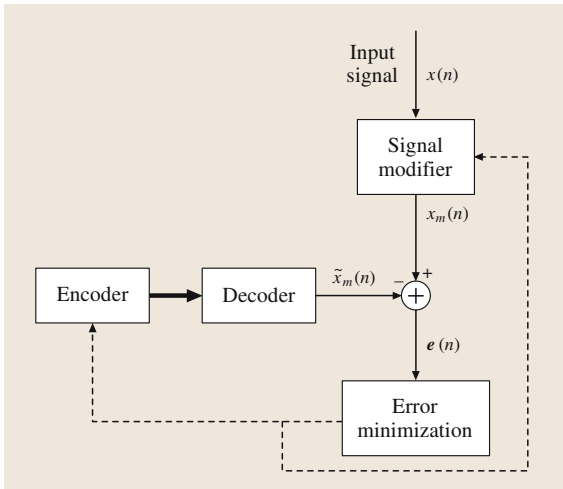


Fig. 17.10 Generalized analysis by synthesis

17.13.1 Generalized Analysis by Synthesis Applied to the Pitch Parameters

Application of generalized analysis-by-synthesis to the pitch parameters implies that one or both of the pitch period contour and the pitch gain contour of the speech signal is modified for easier encoding. A detailed discussion of the application to the pitch period and gain is available in [17.70]. Typically, the pitch period and pitch gain are updated approximately every 5 ms. The idea of RCELP is to update and encode them far less frequently, e.g., every 20 ms, and then use an interpolated pitch period contour and pitch gain contour throughout the 20 ms. This is justified by the observation that the pitch period and periodicity of voiced speech evolve slowly. In order to maintain the coding efficiency when using the interpolated pitch period and gain (compared to the more-frequent update), the speech signal must be modified to follow these contours. Otherwise, the waveform matching of the analysis-by-synthesis principle will break down as the interpolated pitch period and gain will result in a pitch contribution from the adaptive codebook (or equivalently, the pitch predictor) that is misaligned with the reference signal for the analysis-by-synthesis. Since the pitch evolution in voiced speech is slow, typically only minor adjustments to the speech signal are necessary and without impact to the speech quality. This was demonstrated in [17.69] where subjective results showed that the modified speech received mean opinion scores (MOS) very close to those of the original speech, at a level similar to those of 64 kbit/s μ-law (Table 17.4).

Besides exploiting the typical slow evolution of the pitch period to reduce the bits required to encode the pitch period, RCELP can also save the bits otherwise often used to specify fractional pitch lags. Basically, the speech can be modified to fit an interpolated pitch period contour specified by integer lags. Note that the interpolated pitch period contour will have fractional

Table 17.4 MOS of RCELP signal modification [17.69]

	MOS–IRS input	MOS–flat input
64 kb/s μ-law	3.94	4.03
modified speech	3.90	3.99

lags, but at the points of transmission the pitch lag has integer values.

Application to the Pitch Period

Although [17.70] discusses the application of generalized analysis-by-synthesis to both the pitch period and the pitch gain, in speech coding standards, the application to the pitch period has found the widest usage. Accordingly, the application of generalized analysis-by-synthesis to the pitch period is commonly referred as RCELP. In narrow-band speech coding this can bring the bit allocation for the pitch period from $8+5+8+5$ bits to seven bits for a 20 ms frame. This is a significant saving that frees up bits for improving quantization of other parameters of the coder. In practice the pitch period may be estimated open loop and used to create an interpolated continuous pitch track. The speech signal is subjected to minor adjustments in order to fit this interpolated pitch track. Often the adjustments are carried out through time warping and conveniently carried out in the short-term prediction residual domain [17.15]. The modified residual signal can be passed through the short-term synthesis filter or the weighted synthesis filter in order to obtain the modified speech signal or the modified weighted speech signal, respectively. For lower complexity, time shifting of sequential blocks can be applied instead of time warping [17.69]. Later, [17.71] proposed to carry out the signal modification in the weighted speech domain and use a combination of time warping and time shifting of blocks for modifying the speech signal. The method only shifts the high-energy blocks (supposedly containing the pitch pulse and immediate vicinity), while the low-energy blocks are time warped to adjust the pitch period. Effectively, this method does not alter the local waveform properties of the pitch pulses.

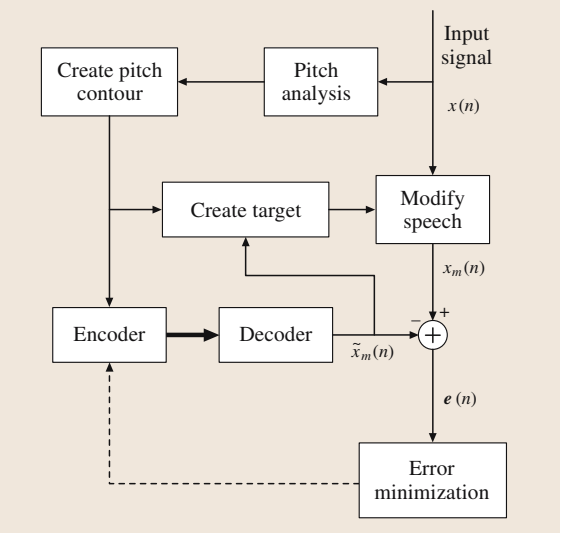


Fig. 17.11 Principle of RCELP

The general sequence of operations for RCELP are:

1. estimate open loop pitch,
2. create pitch contour,
3. create target signal for modifying the speech,
4. modify the speech, and
5. generate the adaptive codebook contribution.

This is depicted in principle at a high level in Fig. 17.11, where the generation of the adaptive codebook contribution takes place inside the encoder like usual, except now according to a pitch contour as opposed to a fixed pitch. Note that the additional blocks in Fig. 17.11 as compared to Fig. 17.9 belong to the encoder just like the *signal modifier* in Fig. 17.10. Furthermore, the decoder would comprise identical functions to

1. create the pitch contour, and
2. generate the adaptive codebook contribution according to the pitch contour.

Note that Fig. 17.11 should be viewed at a conceptual level as the actual signal modification could take place in any domain, but with a corresponding modified input signal.

Estimate Open-Loop Pitch. It is critical for optimal performance of RCELP to get a good estimate of the pitch period. Typically, it is estimated in an open loop manner at the boundary of each frame. The pitch period towards the end of frame m is denoted by $\mathrm{pp}(m)$.

Create Pitch Contour. In the general case where the pitch period of the previous frame, $\mathrm{pp}(m-1)$, and the current frame $\mathrm{pp}(m)$ are relatively close, the pitch period contour is created as a continuous interpolation between the two. A simple continuous interpolation is the linear interpolation between the two pitch periods:

$$\mathrm{ppc}(m,n) = \frac{n+1}{N+1}\mathrm{pp}(m) + \frac{N-n}{N+1}\mathrm{pp}(m-1)\,,$$
$$n = 0, 1, \ldots, N-1\,. \qquad (17.35)$$

The continuous interpolated pitch period contour is referred as the interpolated pitch contour in the following.

Create Target Signal for Modifying the Speech. In order to modify the speech signal to follow the in-

Table 17.5 Properties of RCELP in standards

Standard	RCELP mode	Time-scale search		Time-scale modification		Max. delay drift
		Domain	Resolution	Domain	Method	
EVRC	Continuous	Short-term residual	(1/8)-th sample	Short-term residual	Interpolation	
SMV	Switched	Weighted speech	(1/10)-th sample	Weighted speech	Interpolation & block shifting	±2.5 ms
VMR-WB	Switched	Weighted speech	(1/8)-th sample	Short-term residual	Block shifting	0 ms

terpolated pitch period contour a target signal can be constructed. One possibility is to extrapolate the previously modified speech signal according to the pitch contour and use this as a target for modifying the time-scale of the speech signal of the current frame. Note that in practice, the time-scale modification of the speech signal can take place in any domain. As an example, in EVRC [17.26] it takes place in the short-term residual signal domain, in SMV [17.64] it takes place in the weighted speech signal domain, and in VMR-WB [17.28] the actual time-scale modification is done in the short-term residual signal domain, but the time-scale modification is calculated in the weighted speech signal domain.

Modify the Speech. The speech signal can be modified in multiple ways, and in multiple domains as mentioned above. The goal is to modify the time-scale of the speech signal so that it follows the interpolated pitch contour. With the target signal described above, this can be achieved by modifying the time-scale of the speech signal so as to maximize the correlation between the modified speech signal and the target signal. If the time-scale modification is carried out on a pitch period by pitch period basis, then it becomes a matter of identifying the pitch pulses and perform time-scale modification so that the pitch pulses of the modified speech signal lines up with the pitch pulses of the target signal. The time-scale modification can be carried out either by time warping in the form of resampling, time shifting of blocks, or a combination thereof.

Generate the Adaptive Codebook Contribution. The final step is to generate the adaptive codebook (or equivalently, the pitch predictor) contribution. Based on the past short-term synthesis filter excitation the adaptive codebook contribution is generated by signal interpolation according to the interpolated pitch contour. Fractional resolution in the order of 1/8-th is generally used, and truncated sinc windows are used for the signal interpolation.

A number of practical issues need to be considered when using RCELP:

1. pitch dependence
2. delay drift
3. pitch doubling and halving
4. complexity

Pitch Dependence. Not surprisingly, there is a strong dependency on the pitch period estimation. If the pitch period estimation is not of sufficient accuracy, naturally, degradation in performance should be expected. Furthermore, RCELP works the best for voice speech with slow evolution of the pitch period. Accordingly, some coders have separate mode(s) with more frequent update of the pitch period for frames where a single interpolated pitch contour is not expected to provide satisfactory performance.

Delay Drift. Generally, a variable delay is introduced during the signal modification. This contributes to the overall system delay, and it may drift. Typically, a constraint on the maximum delay drift of about 3 ms is enforced during RCELP. One exception is VMR-WB where the pitch contour is constrained to provide perfect time synchrony between the original speech and the modified speech at frame boundaries. This means that there is no additional delay from RCELP in this implementation.

Pitch Doubling and Halving. Pitch doubling and halving can present a significant challenge if RCELP is used continuously. For multimode coders an alternative mode with more frequent pitch period updates without pitch period interpolation could be used in such cases. However, solutions to continuously use RCELP also for pitch doubling and halving are discussed in [17.70].

Complexity. As reported in [17.70] the complexity of initial versions was very high. Some initial experimental versions were reported to be 75 million operations per second (MOPS), but projections indicated solutions with

a complexity of about 15 MOPS would be achievable. Such solutions have since materialized in the form of standards such as EVRC, SMV, and VMR-WB which are all around 40 weighted MOPS (WMOPS) or less for full duplex encoding and decoding.

17.13.2 RCELP in Standards

RCELP is utilized in a number of standards to efficiently represent the pitch period: TIA-127 EVRC [17.26], TIA/3GPP2 SMV [17.64], 3GPP2 VMR-WB [17.28], and it was part of a proposal for the ITU-T 4 kb/s standardization effort [17.8] before that effort was abandoned by the ITU. Although, those standards implement RCELP in different ways and variations, they all achieve the same goal of reducing the bit rate required to represent the pitch period. Some apply RCELP continuously while others apply RCELP only for specific modes. Some of the RCELP features of these standards are summarized in Table 17.5.

17.14 eX-CELP

The eXtended CELP (eX-CELP) technique [17.71] is more a collection of techniques or a general approach than a single specific technique. The general idea is to emphasize the perceptual important features during encoding within the context of analysis-by-synthesis. Basically, the closed-loop waveform matching of analysis-by-synthesis is relaxed by combining open-loop and closed-loop control. The necessity to relax the strict waveform matching originates from the observation that at low bit rate it is not possible to achieve toll quality through strict waveform matching, and instead emphasis on the perceptually important features is necessary. This is achieved by incorporating signal classification into weighting, using RCELP-like techniques, combining open-loop and closed-loop, designing the fixed codebook with multiple sub-codebooks addressing different signal characteristics, and utilizing multimode encoding.

Signal Classification. eX-CELP uses elaborate signal classification. As many as six classes are used [17.27]:

- silence/background noise
- stationary unvoiced
- nonstationary unvoiced
- onset
- nonstationary voiced
- stationary voiced

The classification takes place in multiple stages and is refined as information becomes available in the algorithm.

Signal Modification. The signal modification algorithm not only modifies the pitch contour to allow a lower bit rate for the encoding of the pitch period like RCELP, it also modifies the speech signal to increase the pitch contribution of the adaptive codebook. According to [17.71], it uses waveform interpolation or harmonic smoothing to pre-smooth voiced transition areas in an open-loop manner. This results in faster buildup of the adaptive codebook. The underlying objective is to increase/enhance the pitch contribution as much as possible without introducing noticeable distortion to the signal.

Combination of Open-Loop and Closed-Loop. The algorithm combines open-loop and closed-loop extensively. One example is the gain quantization. It is well known that for background noise, accurate waveform matching is not necessary for a perceptual accurate reproduction. However, features such as a natural (smooth) energy contour and a dense excitation is more important. Hence, in the presence of background noise the gain quantization favors open-loop over close-loop, and it aims at maintaining a smooth energy contour. Similarly, it would favor selection of excitation from a relatively denser codebook.

Fixed Codebook. The fixed codebook is made up of multiple sub-codebooks of varying pulse density and even a sub-codebook of dense random-like noise. Each sub-codebook is designed and tuned with a certain type of signal or speech in mind. Each codebook is generally searched according to the analysis-by-synthesis principle, but the selection between the best outputs of the various sub-codebooks is influenced by the classification information, and other signal parameters such as estimate of background noise level and *peakiness* of the speech.

Multimode. eX-CELP distinguishes between two fundamentally different encoding modes, types 0 and 1.

Table 17.6 Key general features of type 0 and type 1 coding modes in eX-CELP

Entity	Type 0	Type 1
Signal character	All other	Staitonary voiced
Subframes	Fewer	More
Pitch period (adaptive codebook)	Subframe based	Frame based
Fixed codebook	Pulse & Gaussian sub-codebooks	Pulse sub-codebooks
Adaptive codebook gains	Joint 2-D VQ (subframe based)	A priori VQ (frame based)
Fixed codebook gains	Joint 2-D VQ (subframe based)	A posteriori VQ (frame based)

Table 17.7 MOS of SMV at various operating points [17.71]

Coder	Average bit rate	MOS (clean speech)	MOS (noisy speech)
EVRC	x	3.58	3.35
SMV Mode 0	$1.0 \cdot x$	3.90	3.57
SMV Mode 1	$0.71 \cdot x$	3.64	3.53
SMV Mode 2	$0.56/0.60 \cdot x$	3.46	3.53

Table 17.8 MOS of 4 kbit/s eX-CELP [17.8]

Input	Coder	MOS
MIRS (modified intermediate reference system)	G.726	3.14
	G.729	3.43
	4 kb/s eX-CELP	3.36
Flat	G.726	3.12
	G.729	3.27
	4 kb/s eX-CELP	3.32

The bit allocation, the quantization methods, and even the subframe structure for the two modes can be different. In type 1, targeting stationary voiced speech, the subframe adaptive codebook gains are prequantized open-loop using VQ prior to subframe processing, while fixed codebook gains are left unquantized during subframe processing and quantized jointly using VQ after subframe processing is complete. For type 0, a more-conventional joint two-dimensional VQ of the subframe based adaptive and fixed codebook gains is used. However, the gain estimation and quantization is still not strictly analysis-by-synthesis for type 0, as focus remains on producing a perceptually faithful output, mixing open-loop and closed-loop. Key general features of the two types are listed in Table 17.6.

17.14.1 eX-CELP in Standards

The 3GPP2 SMV standard [17.64] is based on eX-CELP. Table 17.7 is reproduced from [17.71] and compares the MOS scores of SMV at various operating points of quality versus average bit rate with EVRC. The third column contains the MOS scores for clean speech, and the fourth column contains the MOS scores of speech in background noise. Also one of the promising ITU-T 4 kbit/s candidates [17.8] was based on eX-CELP. Some MOS scores comparing the performance to 32 kbit/s G.726 and 8 kbit/s G.729 are reproduced in Table 17.8.

17.15 iLBC

The iLBC coder [17.31, 32] is quite different from other analysis-by-synthesis speech coders. Most other analysis-by-synthesis speech coders use a long-term predictor or an adaptive codebook continuously and across the frame boundaries. Due to the relatively large bulk delay in repeating the previous waveform stored in the long-term predictor memory or the adaptive codebook, if a frame is lost, the degrading effect tends to propagate a while. The iLBC coder attempts to address this issue and improve the robustness against frame loss. It achieves this by employing block-independent coding of the adaptive codebook. In other words, the adaptive codebook of each speech frame is encoded independent of the previous frames. Therefore, if a frame is lost, the quality degrading effect due to a mismatched adaptive codebook is limited to the lost frame and will not propagate to the future frames.

Of course, the improved robustness does not come for free. By not allowing the adaptive codebook to be used across the frame boundaries, the iLBC coder gives up the opportunity to exploit the corresponding long-term redundancy in the speech signal across the frame boundaries. Therefore, for clear channel it inevitably sacrifices the output speech quality to some extent when compared with more-conventional CELP coders. In effect, the iLBC coder makes a trade-off between the

speech quality in clear-channel and the speech quality under frame loss conditions. Given that the iLBC coder was developed for voice over internet protocol (VoIP) over the general Internet that can have a fairly high packet loss rate, it is reasonable for iLBC to make such a trade-off.

The iLBC coder was standardized as an IETF experimental standard [17.32]. It has two versions: a 13.3 kb/s version with a 30 ms frame size and 10 ms look-ahead, and a 15.2 kb/s version with a 20 ms frame size and 5 ms look-ahead. The two versions divide each speech frame into 5 ms subframes. The short-term prediction residual signal is first calculated. The two consecutive subframe of residual having the largest weighted energy are identified. Within these two subframes, the *start state (segment)* is selected as either the first S samples or the last S samples of the two consecutive subframes, depending on which segment has a higher energy. The integer S is 57 or 58 samples, depending on whether it is the 20 ms version of iLBC or the 30 ms version. The adaptively selected start state is encoded with scalar quantization without using information in the previous frames.

A dynamic codebook encoding procedure first encodes the remaining samples in the two subframes containing the start state. Next, it encodes the remaining subframes forward in time, and then it encodes the remaining subframes backward in time. The maximum-energy selection criterion allows iLBC to capture the pitch epoch of a pitch cycle waveform or the onset of voiced segments as the start state. The forward–backward approach mentioned above then allows an adaptive codebook to be constructed and used in the entire current frame without using the adaptive codebook in the previous frames.

There are further details in the codebook search method and how iLBC encodes the codebook gains and re-scale the gain for power matching after encoding. Interested readers are referred to [17.32].

It should be noted that even though iLBC performs frame-independent coding of the adaptive codebook, the iLBC coder is not completely frame independent. At least the short-term synthesis filter memory from the previous frame is still used when starting the encoding and decoding operation in the current frame. This means that iLBC still has a slight error propagation from one frame to the next, although the degree of error propagation is significantly smaller than that of the other conventional CELP coders.

17.16 TSNFC

Two-stage noise feedback coding (TSNFC) [17.72] is a relatively new class of analysis-by-synthesis coders although it builds on the decade-old technique of noise feedback coding (NFC) [17.16, 73]. The ANSI Amer-

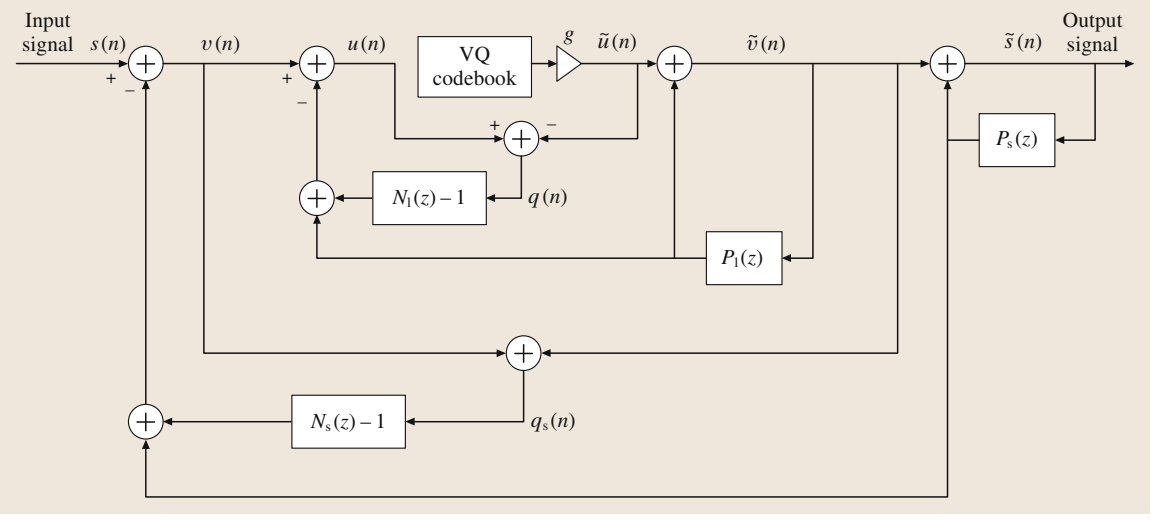


Fig. 17.12 TSNFC encoder structure

ican national standard BV16 coder [17.33] and the similar wide-band version BV32 are based on TSNFC. While the original NFC uses sample based quantization of the excitation signal and has less in common with analysis-by-synthesis, the recent extension of NFC to TSNFC and the development of CELP-like VQ techniques [17.72, 74] for vector quantization of the excitation signal have created a commonality between TSNFC and CELP, making a brief overview in the context of analysis-by-synthesis relevant.

One structure for TSNFC is shown in Fig. 17.12. It clearly looks nothing like a CELP coder, compare to Fig. 17.5. The short-term predictor, $P_s(z)$, is equivalent to the short-term predictor of CELP coders and results in a short-term prediction error filter given by

$$A(z) = 1 - P_s(z)\,. \tag{17.36}$$

Similarly, the long-term predictor $P_l(z)$ corresponds to the pitch predictor of CELP (or the adaptive codebook). Before TSNFC is described in more detail another interesting observation in relation to analysis-by-synthesis can be made. Studying the TSNFC encoder structure in Fig. 17.12 one can observe the decoder output speech, $\tilde{s}(n)$, being generated as part of the quantization process of the excitation. Conceptually, every codevector of the VQ codebook is fed through the structure in Fig. 17.12 and the candidate minimizing the MSE of $q(n)$ is selected as the output of the VQ. With a short-term noise feedback filter given by $N_s(z) - 1$, minimizing the MSE of $q(n)$ will result in a spectral envelope of the quantization noise, $e(n) = \tilde{s}(n) - s(n)$ given by $N_s(z)$. Hence, the spectral envelope of the coding noise can be controlled directly by $N_s(z)$. This has a parallel to the CELP coder where the spectral envelope of the coding noise is controlled by the inverse of the perceptual weighting filter. Hence, a CELP coder with a perceptual weighting filter of

$$W(z) = \frac{A(z/\gamma_1)}{A(z/\gamma_2)}\,, \quad 0 < \gamma_2 < \gamma_1 < 1\,, \tag{17.37}$$

and a TSNFC coder with a short-term noise feedback filter of

$$N_s(z) - 1 = \frac{A(z/\gamma_2)}{A(z/\gamma_1)} - 1\,, \quad 0 < \gamma_2 < \gamma_1 < 1 \tag{17.38}$$

will both result in a coding noise with a spectral envelope shaped according to

$$N_s(z) = \frac{A(z/\gamma_2)}{A(z/\gamma_1)}\,. \tag{17.39}$$

Similarly, the TSNFC structure in Fig. 17.12 will result in a coding noise with spectral fine structure shaped according to $N_l(z)$. A suitable choice according to [17.33] is

$$N_l(z) = 1 + \beta z^{-T}\,, \tag{17.40}$$

where β is the long-term noise feedback filter coefficient and T is the pitch period. Typically, β is adaptively controlled by the pitch analysis, and $0 \le \beta \le 1$.

Derivation of short-term predictor coefficients and quantization thereof can be carried out with appropriate techniques from the literature, similarly with the estimation and quantization of the long-term predictor parameters. The methods and structures for VQ of the excitation and the similarities to analysis-by-synthesis in CELP coders will be discussed in more detail in the following.

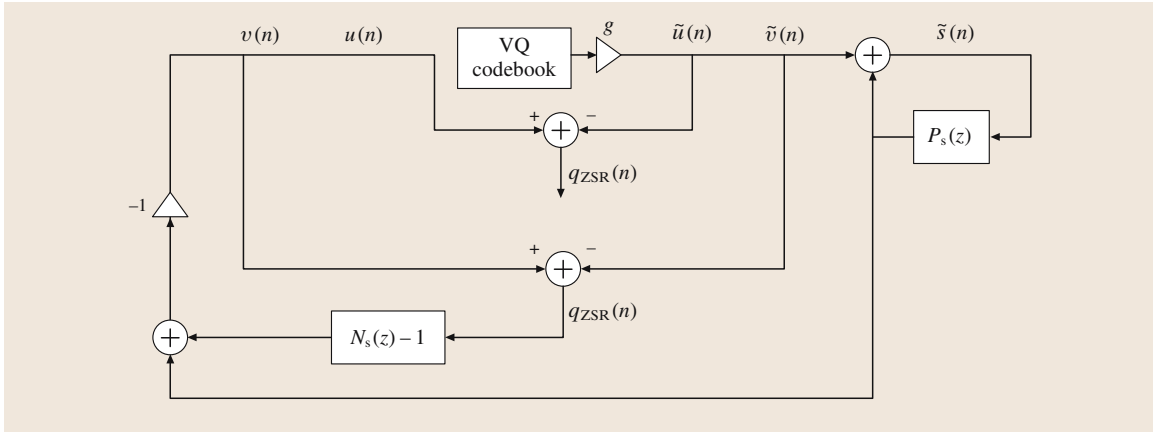


Fig. 17.13 TSNFC ZSR structure

17.16.1 Excitation VQ in TSNFC

Similar to CELP it is advantageous to apply the superposition principle when calculating the error signal, $q(n)$, for the excitation VQ. It is calculated as the superposition of the zero state response (ZSR), $q_{\mathrm{ZSR}}(n)$, and the zero-input response (ZIR), $q_{\mathrm{ZIR}}(n)$ [17.72]. With a minimum pitch lag being greater than the dimension of the excitation VQ, the structure in Fig. 17.12 reduces to the ZSR structure in Fig. 17.13. This structure is simplified as shown in Fig. 17.14 according to [17.74], where $H(z)$ with the short-term noise feedback filter of (17.38) is given by

$$H(z) = \frac{Q_{\mathrm{ZSR}}(z)}{\tilde{U}(z)} = -\frac{1}{N_{\mathrm{s}}(z)A(z)} = -\frac{A(z/\gamma_1)}{A(z/\gamma_2)A(z)} \,. \tag{17.41}$$

The ZIR structure is derived from Fig. 17.12 as shown in Fig. 17.15. Typically in TSNFC, a relatively short dimension of the excitation VQ is used. In BV16 [17.33] a 4-dimensional excitation VQ is used. The frame size is 40 samples (5 ms) during which $A(z)$ remains constant. Hence, the codebook needs only to be filtered with $H(z)$ according to Fig. 17.14 once per frame, or equivalently, once per 10 codebook searches. In other words, the filtered codebook is generated only once per 10 codebook searches. The filtered codebook vectors are denoted $q_{\mathrm{ZSR}}(k, n)$, where k indicates the codebook index, $k = 0, 1, \ldots, K-1$. On the other hand, the ZIR, $q_{\mathrm{ZIR}}(n)$ needs to be generated for every four-sample excitation vector. However, it is independent of the codebook vectors. Hence, on a frame basis:

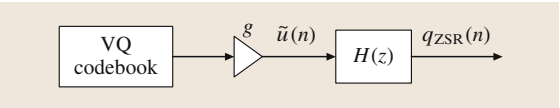


Fig. 17.14 Simplified TSNFC ZSR structure

- the filtered codebook vectors need to be generated once,
- the 10 ZIR vectors needs to be generated, once for each four sample excitation vector,
- and after each of the 10 codebook searches and before the next ZIR calculation the filter memories need to be updated.

Efficient methods for updating the filter memory is presented in [17.74]. The codebook search is carried out efficiently according to

$$k_{\mathrm{opt}} = \arg\min_k \left\{ \sum_{n=0}^{N-1} [q_{\mathrm{ZIR}}(n) + q_{\mathrm{ZSR}}(k, n)]^2 \right\} \,, \tag{17.42}$$

where $N = 4$ in the example above. With a signed codebook as in BV16, and using vector form notation, this reduces to

$$k_{\mathrm{opt}} = \arg\min_k \left\{ E\left[\boldsymbol{q}_{\mathrm{ZSR}}(k)\right] \overset{s}{\pm} R\left[\boldsymbol{q}_{\mathrm{ZSR}}(k), \boldsymbol{q}_{\mathrm{ZIR}}\right] \right\} \,, \tag{17.43}$$

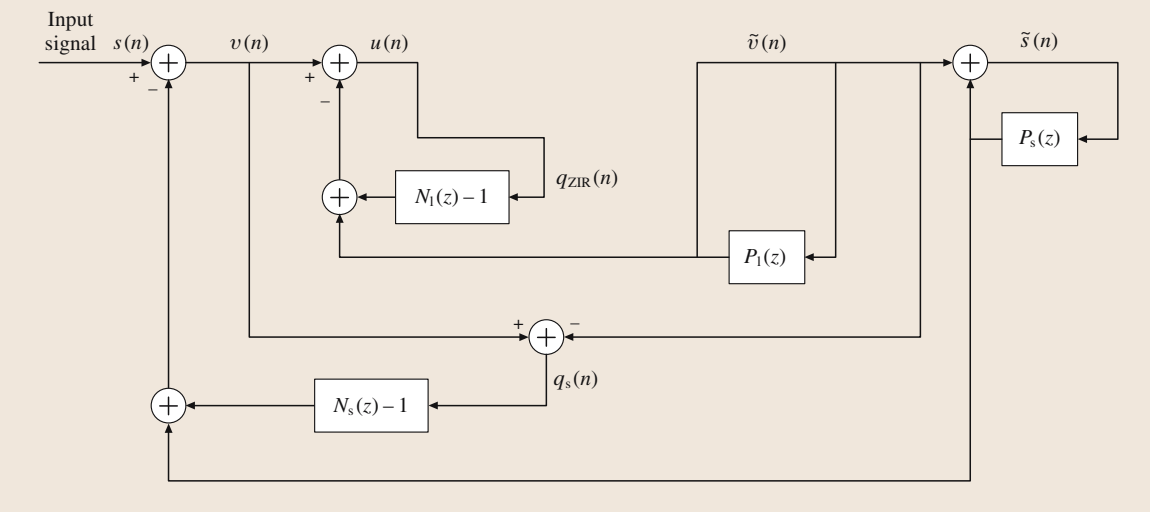


Fig. 17.15 TSNFC ZIR structure

Table 17.9 MOS of BV16

Coder	MOS
G.711	3.91
G.726	3.56
G.728	3.54
G.729	3.56
BV16	3.76

Table 17.10 MOS of BV32

Coder	MOS
G.722 @ 48 kb/s	3.60
G.722 @ 56 kb/s	3.88
G.722 @ 64 kb/s	3.96
BV32	4.11

where s indicates the sign, and

$$E\left[\boldsymbol{q}_{\mathrm{ZSR}}(k)\right] = \sum_{n=0}^{N-1} q_{\mathrm{ZSR}}^2(k,n)$$

$$R\left[\boldsymbol{q}_{\mathrm{ZSR}}(k), \boldsymbol{q}_{\mathrm{ZIR}}\right] = \sum_{n=0}^{N-1} q_{\mathrm{ZSR}}(k,n) \cdot q_{\mathrm{ZIR}}(n)\,.$$

Note that the optimal sign is given as the opposite sign of the correlation, and hence, effectively, only half the codevectors need to be searched. Similarly, with a signed codebook only half the codevectors need to be filtered with $H(z)$ as the other half of filtered codevectors are given by simple negation. In (17.43), $E\left[\boldsymbol{q}_{\mathrm{ZSR}}(k)\right]$ can be pre-computed before the first of the 10 excitation vectors is quantized. Hence, the only search-loop computation ends up being the calculation of $R\left[\boldsymbol{q}_{\mathrm{ZSR}}(k), \boldsymbol{q}_{\mathrm{ZIR}}\right]$. These techniques are very similar to techniques used in CELP coding, but now applied to the TSNFC structure.

17.16.2 TSNFC in Standards

The PacketCable, SCTE, and ANSI BV16 standard for voice over cable is based on TSNFC. BV16 is a narrow-band coder and has an algorithmic delay of 5 ms, a bit rate of 16 kbit/s, a complexity comparable to G.729A (significantly lower than G.729 and G.728). Results from a formal subjective test of BV16 are summarized in Table 17.9. CableLabs has also included BV32 [17.30], a wide-band version of BV16, in PacketCable 2.0 codec and media specification [17.75]. BV32 also has an algorithmic delay of 5 ms, a bit-rate of 32 kbit/s, a complexity comparable to G.729 (significantly lower than G.722.2/AMR-WB), and a subjective quality better than 64 kbit/s G.722. Formal subjective test results of BV32 are summarized in Table 17.10.

17.17 Embedded CELP

The recently standardized ITU-T G.729.1 coder [17.76] is a bit-rate- and bandwidth-scalable embedded coder based on G.729. The 16 kHz sampled wide-band input speech is split into equal-bandwidth high and low bands with a quadrature mirror filterbank (QMF). The low-band (narrow-band) signal is encoded with the narrow-band core (also referred as Layer 1) layer. Although the frame size of G.729.1 is 20 ms as opposed to the 10 ms frame size of G.729, the core layer is bitstream compatible with G.729. However, G.729.1 has a significantly longer algorithmic delay. The algorithmic delay of G.729 is 15 ms while the algorithmic delay of G.729.1 is 48.9375 ms. Layer 2 is a 4 kb/s narrow-band embedded enhancement layer to G.729. Layers 3–12 provide wide-band (50–7000 Hz) capability at a total bit rate increasing from 14 kbit/s to a maximum of 32 kb/s at increments of 2 kb/s per layer. An overview of the encoder is presented in Fig. 17.16. Layer 3 encodes the high-band signal using time-domain bandwidth extension (TDBWE) [17.76]. Layers 4–12 encode the weighted error signal from the narrow-band layer 1 and 2 embedded CELP coder, 50–4000 Hz, jointly with the high-band signal, 4000–7000 Hz. A transform predictive coder (TPC) with time-domain aliasing cancelation (TDAC) is utilized for layers 4–12. The coder sends redundant information to the decoder in order to mitigate

Table 17.11 Overview of G.729.1 embedded CELP

Layer	Technology	Analysis by synthesis	Bandwidth	Cummulative bit rate
1	ACELP	Yes	50–4000 Hz	8 kb/s
2	ACELP	Yes	50–4000 Hz	12 kb/s
3	TDBWE	No	4000–7000 Hz	14 kb/s
4–12	TPC	No	50–7000 Hz	16–32 kb/s

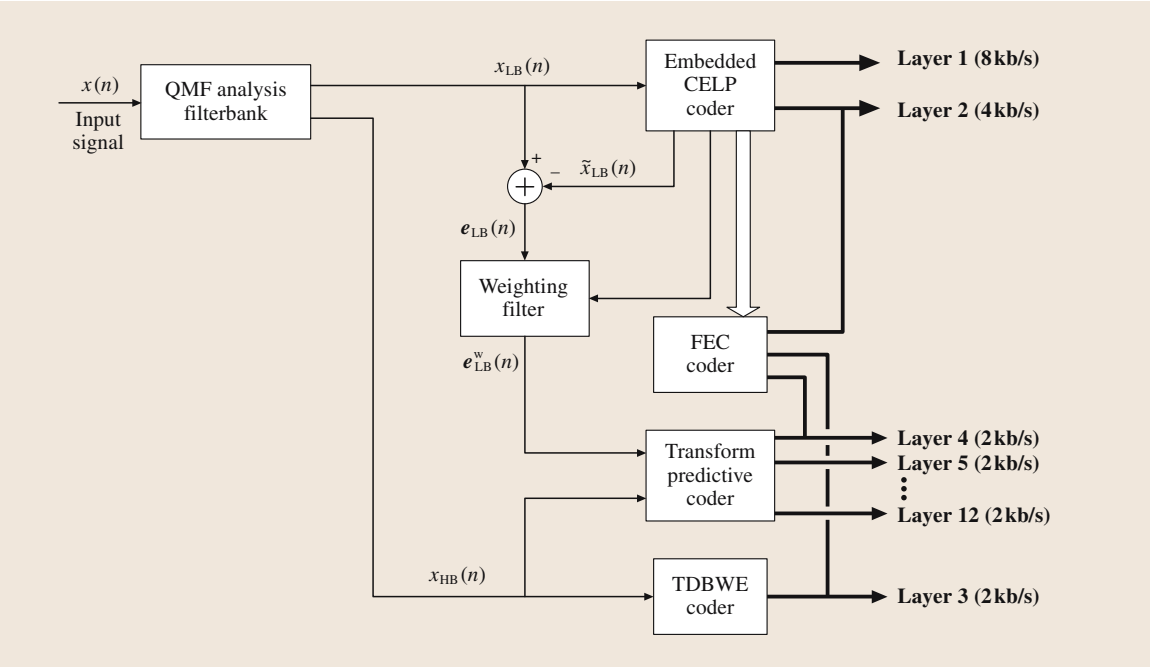


Fig. 17.16 High-level block diagram of G.729.1

the effect of frame loss. This information is calculated with the frame erasure concealment (FEC) encoder and comprises information pertaining to the narrow-band embedded CELP coder in order to speed up recovery of the CELP decoder after frame loss. The information includes

- signal classification information
- estimated position of the last glottal pulse
- energy calculated as either maximum or average sample energy (dependent on signal classification)

Table 17.12 elaborates the FEC encoded information with bit-allocation and information regarding which layer includes the additional information. It should be noted that the FEC information is only available if the layers listed in Table 17.12 are actually received by the decoder.

Table 17.12 Redundant information included in G.729.1 bitstream to help embedded CELP coder recover from frame loss

Information	Location	Bit allocation
Classification	Layer 2	2 bits
Glottal pulse location	Layer 3	7 bits
Energy	Layer 4	5 bits

Since the TDBWE and transform predictive coders are not analysis-by-synthesis coders, they will not be treated in any greater details here. Instead, the reader is referred to [17.76] for further information.

The embedded CELP coder is, however, of interest as not only the core layer, but also the enhancement layer, is derived according to analysis-by-synthesis and presents a particular problem. In the following, the terms encoder and decoder are used to refer to the embedded CELP only encoder and decoder portions of G.729.1. The particular problem pertains to the issue of ensuring that the encoder and decoder remain synchronized regardless of the number of decoded layers. Consider the fundamental meaning of the term analysis-by-synthesis. Regardless of the number of decoded layers the coder must remain an analysis-by-synthesis coder:

At the encoder, without knowing how many layers the decoder will decode, how is analysis-by-synthesis maintained?

This appears to be an oxymoron, but the key is to realize that the encoder has multiple layers of encoding, and then structure the algorithm such that the various levels of encoding remain synchronized with the various levels of decoding. This may sound simple. However, looking at an example, it quickly becomes clear that it is not quite that simple. Let us assume that

a CELP enhancement layer includes a fixed codebook. The straightforward way would be to update the adaptive codebook at the encoder with a short-term excitation signal including this enhancement layer contribution. This would be optimal in the sense of the output speech quality including the enhancement layer. However, at the decoder this would present a problem if the enhancement layer is not received as the adaptive codebook would then no longer be synchronized with the encoder. The analysis-by-synthesis property would be violated: in retrospect the encoder did not do the same synthesis part in the analysis-by-synthesis process as the decoder. Once the analysis-by-synthesis property is violated the speech quality is somewhat up in the air, and the waveform matching during analysis-by-synthesis is somewhat senseless as the decoder will not repeat the same synthesis.

Not only does the example illustrate the particular issue of embedded analysis-by-synthesis, it also demonstrates one reason why an embedded analysis-by-synthesis coder would be of inferior speech quality to a nonembedded analysis-by-synthesis coder: the obvious optimal approach, in terms of optimizing the speech quality of the output including the enhancement layer, had to be disregarded in order not to violate the analysis-by-synthesis property.

To maintain the analysis-by-synthesis property for embedded analysis-by-synthesis, the memory of one layer cannot be updated with any information of a higher layer. In the example above, with an enhancement layer containing an additional fixed codebook, it means that the adaptive codebook of the lower layer (the core layer) can only be updated with the short-term synthesis filter excitation without the fixed codebook contribution of the enhancement layer. Furthermore, at the encoder, separate short-term synthesis filter memory may need to be maintained. Another example, albeit perhaps not truly analysis-by-synthesis, is the ADPCM encoder of G.726 [17.77] and the low-band ADPCM encoder of G.722 [17.78]. Both are embedded backward adaptive predictive coders. In G.722 the low-band prediction error signal is quantized sample by sample with six bits. However, the low-band ADPCM decoder can decode based on four, five, or six bits per sample. In order for the backward adaptive entities of the low-band ADPCM to stay synchronized between encoder and decoder at all times, the encoder and decoder only use the four-bit information for updating the backward adaptive entities of the low-band ADPCM.

17.18 Summary of Analysis-by-Synthesis Speech Coders

Table 17.13 lists and compares the properties of the various analysis-by-synthesis techniques and coders. Each of the Sects. 17.4 through 17.17, with detailed discussions of the various analysis-by-synthesis techniques, summarize which standards utilize the techniques. This section will view it from a different angle, and for individual speech coding standards list the applicable techniques and features. Note that multiple techniques apply to some of the standards. Furthermore, key characteristics of the coders are provided as well in order to put the techniques into context and provide the reader with a general overview of properties of speech coding standards. It should be noted that the scope was limited to analysis-by-synthesis coders, and that space limitations prevent inclusion of every standard speech coder. Note that ACB for 'pitch type' indicates the use of the adaptive codebook approach to implement the pitch prediction, while 'filter' indicates implementation as a regular filter. The term 'frac. lag' indicates usage of fractional pitch lag. Furthermore, 'excitation' refers to the fixed codebook (or innovation) excitation, and hence, excludes the pitch related part of the excitation. The bit rate for the 'excitation' in Table 17.13 also excludes the gain for the fixed codebook excitation. The column 'year standardized' should only be considered approximate, as there is some variation in the years listed in the literature due to a common delay between the initial selection of a speech coding algorithm, and the date it is officially standardized. Furthermore, the 'complexity' should only be considered a rough guideline as great variations for a given algorithm exist. Such variation can be due to many factors, e.g., capability of target processor, level of optimization, etc. Also, it should be noted that the unit for the complexity is either WMOPS or MIPS. The WMOPS number is typically obtained from a simulation software in C code with operators with weights emulating the complexity of each operator on a typical digital signal processor (DSP). On the other hand, the MIPS number is typically obtained from an actual implementation on a specific DSP. Although the purpose of the WMOPS number is to estimate the complexity on a DSP (the MIPS number), it does not always provide an exact number.

Table 17.13 Overview of analysis-by-synthesis speech coding standards

Coder	Year standardized	Standards organization	Bit rate	Audio bandwidth	Algorithmic delay	Complexity	Excitation type	Excitation dimension	Excitation bit rate	Pitch type	Other AbS techniques
GSM-FR	1986	ETSI	13 kb/s	0–4 kHz	20 ms	4.5 MIPS	RPE	40	8.2 kb/s	Filter	
FS-1016	1989	DoD	4.8 kb/s	0–4 kHz	30 ms	25 MIPS	Overlap sparse	60	1.33 kb/s	ACB	
IS-54	1989	TIA	8 kb/s	0–4 kHz	28.125 ms	25 MIPS	VSELP	40	2.8 kb/s	ACB	
G.728	1992	ITU-T	16 kb/s	0–4 kHz	0.625 ms	36 MIPS	Trained	5	11.2 kb/s	None	LD-CELP
GSM-HR	1993	ETSI	5.6 kb/s	0–4 kHz	25 ms	25 MIPS	VSELP	40	1.8 kb/s 2.8 kb/s	ACB fraction lag	
PDC-HR	1993	ARIB	3.45 kb/s	0–4 kHz	45 ms	18.7 MOPS	PSI	80		ACB fraction lag	
EVRC	1994	TIA	8.55 kb/s 4.0 kb/s	0–4 kHz	33 ms	25 MIPS	ACELP	53/43	5.25 kb/s 1.5 kb/s	ACB	RCELP
G.723.1	1995	ITU-T	6.3 kb/s 5.3 kb/s	0–4 kHz	37.5 ms	19 MIPS	MP ACELP	60	3.3 kb/s 2.27 kb/s	ACB 5-tap	
G.729	1996	ITU-T	8 kb/s	0–4 kHz	15 ms	22 MIPS	ACELP	40	3.4 kb/s	ACB fraction lag	CS-ACELP
TIA-EFR	1996	TIA	7.4 kb/s	0–4 kHz	25 ms	14 WMOPS	ACELP	40	3.4 kb/s	ACB fraction lag	
PDC-EFR	1996	ARIB	6.7 kb/s	0–4 kHz			ACELP	40			
GSM-EFR	1997	ETSI	12.2 kb/s	0–4 kHz	20 ms	18 MIPS	ACELP	40	7 kb/s	ACB fraction lag	
GSM-AMR	1999	ETSI	4.75 kb/s to 12.2 kb/s	0–4 kHz	25 ms	20 MIPS	ACELP	40	1.8 kb/s to 7 kb/s	ACB fraction lag	
SMV	2001	3GPP2	8.55 kb/s 4.0 kb/s	0–4 kHz	30.5–35.5 ms 27.5–32.5 ms	40 WMOPS	Sub ACELP	40 80, 53/54	4.4, 6 kb/s 1.5, 1.95 kb/s	ACB	eX-CELP RCELP
AMR-WB G.722.2	2001	3GPP ITU-T	6.6 kb/s to 23.85 kb/s	0–8 kHz	26 ms	38 WMOPS	ACELP	64	2.4–17.6 kb/s	ACB fraction lag	
VMR-WB	2002	3GPP2	6.6 kb/s 8.85 kb/s 12.65 kb/s	0–8 kHz	33.75 ms	38 WMOPS	ACELP	64	2.4–7.2 kb/s	ACB fraction lag	RCELP
iLBC	2002	IETF PacketCable	13.3 kb/s 15.2 kb/s	0–4 kHz	40 ms 25 ms	18 MIPS 15 MIPS	FB-LPC	40	12.0 kb/s 14.2 kb/s	FB-ACB	
BV16	2003	SCTE ANSI	16 kb/s	0–4 kHz	5 ms	12 MIPS	Trained	4	10 kb/s	Filter 3-tap	TSNFC
G.729.1	2006	ITU-T	8 kb/s to 32 kb/s	0–4 kHz 0–8 kHz	48.9375 ms	36 WMOPS	ACELP	40	3.4 kb/s+ 3.4 kb/s	ACB fraction lag	Embedded CS-CELP

17.19 Conclusion

The analysis-by-synthesis method for coding the excitation signal of linear predictive coders has been the most important driving force behind the relentless reduction of the bit-rate for high-quality speech coding in the last two decades. Almost all speech coding standards established after 1989 are based on it. This chapter gives a tutorial on analysis-by-synthesis speech coding techniques in general and describes many variations of the analysis-by-synthesis excitation coding paradigm in particular. Due to space limitation, the description concentrates on dominant types of analysis-by-synthesis excitation coding techniques as exemplified by various speech coding standards, with the relationship between them discussed in the context of a family tree. It is hoped that, after reading this chapter, the reader will have a good understanding of the dominant types of analysis-by-synthesis speech coding techniques.

References

17.1 R.V. Cox: Speech coding standards. In: *Speech Coding and Synthesis*, ed. by W.B. Kleijn, K.K. Paliwal (Elsevier Science, Amsterdam 1995)

17.2 B.S. Atal, J.R. Remde: A new model of LPC excitation for producing natural-sounding speech at low bit rates, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1982) pp. 614–617

17.3 B.S. Atal, M.R. Schroeder: Stochastic coding of speech signals at very low bit rates, Proc. IEEE Int. Conf. Commun. (1984) p. 48.1

17.4 M.R. Schroeder, B.S. Atal: Code-excited linear prediction (CELP): high quality speech at very low bit rates, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1985) pp. 937–940

17.5 J.-H. Chen: A robust low-delay CELP speech coder at 16 kbit/s, Proc. IEEE Global Commun. Conf. (1989) pp. 1237–1241

17.6 J.-H. Chen, R.V. Cox, Y.-C. Lin, N.S. Jayant, M.J. Melchner: A low-delay CELP coder for the CCITT 16 kb/s speech coding standard, IEEE J. Sel. Areas Commun. **10**(5), 830–849 (1992)

17.7 R. Salami, C. Laflamme, J.-P. Adoul, A. Kataoka, S. Hayashi, T. Moriya, C. Lamblin, D. Massaloux, S. Proust, P. Kroon, Y. Shoham: Design and description of CS-ACELP: a toll quality 8 kb/s speech coder, IEEE Trans. Speech Audio Process. **6**(2), 116–130 (1998)

17.8 J. Thyssen, Y. Gao, A. Benyassine, E. Shlomot, C. Murgia, H.-Y. Su, K. Mano, Y. Hiwasaki, H. Ehara, K. Yasunaga, C. Lamblin, B. Kovesi, J. Stegmann, H.-G. Kang: A candidate for the ITU-T 4 kbit/s speech coding standard, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (2001) pp. 681–684

17.9 A. McCree, J. Stachurski, T. Unno, E. Ertan, E. Paksoy, R. Viswanathan, A. Heikkinen, A. Ramo, S. Himanen, P. Blocher, O. Dressler: A 4 kb/s hybrid MELP/CELP speech coding candidate for ITU standardization, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (2002) pp. 629–632

17.10 P. Kroon, W.B. Kleijn: Linear-prediction based analysis-by-synthesis coding. In: *Speech Coding and Synthesis*, ed. by W.B. Kleijn, K.K. Paliwal (Elsevier Science, Amsterdam 1995)

17.11 M. Halle, K.N. Stevens: Analysis by synthesis, Proc. Sem. Speech Compression and Process., Vol. II, ed. by W. Wathen-Dunn, L.E. Woods (1959), AFCRC-TR-59-198, Paper D7

17.12 C.G. Bell, H. Fujisaki, J.M. Heinz, K.N. Stevens, A.S. House: Reduction of speech spectra by analysis-by-synthesis techniques, J. Acoust. Soc. Am. **33**(12), 1725–1736 (1961)

17.13 N.S. Jayant, P. Noll: *Digital Coding of Waveforms* (Prentice Hall, Englewood Cliffs 1984)

17.14 L.R. Rabiner, R.W. Schafer: *Digital Processing of Speech Signals* (Prentice Hall, Englewood Cliffs 1978)

17.15 W.B. Kleijn, R.P. Ramachandran, P. Kroon: Generalized analysis-by-synthesis coding and its application to pitch prediction, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1992) pp. I–337–I–340

17.16 B.S. Atal, M.R. Schroeder: Predictive coding of speech signals and subjective error criteria, IEEE Trans. Acoust. Speech Signal Process. **3**, 247–254 (1979)

17.17 I.A. Gerson, M.A. Jasiuk: Vector sum excited linear prediction (VSELP) speech coding at 8 kbps, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1990) pp. 461–464

17.18 P. Vary, K. Hellwig, R. Hofmann, R.J. Sluyter, C. Galand, M. Rosso: Speech codec for the European mobile radio system, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1988) pp. 227–230

17.19 S. Singhal, B. Atal: Improving performance of multi-pulse LPC coders at low bit rates, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1984) pp. 9–12

17.20 J.P. Campbell Jr., V.C. Welch, T.E. Tremain: An expandable error-protected 4800 bps CELP coder

(U.S. Federal standard 4800 bps voice coder), Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1989) pp. 735–738
17.21 I.A. Gerson, M.A. Jasiuk: A 5600 bps VSELP speech coder candidate for half-rate GSM, Proc. 1993 IEEE Workshop Speech Coding for Telecommunications (1993) pp. 43–44
17.22 T. Ohya, H. Suda, T. Miki: 5.6 kbits/s PSI-CELP of the half-rate PDC speech coding standard, Proc. 1994 IEEE Vehicular Technol. Conf. (1994) pp. 1680–1684
17.23 J.-P. Adoul, P. Mabilleau, M. Delprat, S. Morisette: Fast CELP coding based on algebraic codes, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1987) pp. 1957–1960
17.24 C. Laflamme, J.-P. Adoul, H.Y. Su, S. Morisette: On reducing computational complexity of codebook search in CELP coder through the use of algebraic codes, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1990) pp. 177–180
17.25 ITU-T: G.723.1: Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s (1996)
17.26 ANSI/TIA-127-A-2004: Enhanced variable rate codec speech service option 3 for wideband spread spectrum digital systems (2004) (ANSI/TIA-127-A-2004)
17.27 Y. Gao, E. Shlomot, A. Benyassine, J. Thyssen, H. Su: The SMV algorithm selected by TIA and 3GPP2 for CDMA applications, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (2001) pp. 709–712
17.28 3GPP2 C.S0052-0 V1.0: Source-controlled variable-rate multimode wideband speech codec (VMR-WB) service option 62 for spread spectrum systems, (June 11 2004)
17.29 J.-H. Chen, J. Thyssen: BroadVoice16: A PacketCable speech coding standard for cable telephony, Proc. 40th Asilomar Conf. Signals Systems and Computers (2006)
17.30 J.-H. Chen, J. Thyssen: The BroadVoice speech coding algorithm, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 4 (2007) p. IV-549-IV-552
17.31 S.V. Andersen, W.B. Kleijn, R. Hagen, J. Linden, M.N. Murthi, J. Skoglund: iLBC – a linear predictive coder with robustness to packet losses, Proc. IEEE Workshop Speech Coding (2002) pp. 23–25
17.32 S.V. Andersen, A. Duric, H. Astrom, R. Hagen, W.B. Kleijn, J. Linden: Internet low bit rate codec (iLBC), IETF RFC **3951** (2004)
17.33 American National Standard: BV16 speech codec specification for voice over ip applications in cable telephony, ANSI/SCTE 24-21 2006 (2006)
17.34 B.S. Atal, S.L. Hanauer: Speech analysis and synthesis by linear prediction, J. Acoust. Soc. Am. **50**, 637–655 (1971)
17.35 E.F. Deprettere, P. Kroon: Regular excitation reduction for effective and efficient LP-coding of speech, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1985) pp. 965–968
17.36 W.B. Kleijn, D.J. Krasinski, R.H. Ketchum: Improved speech quality and efficient vector quantization in SELP, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1988) pp. 155–158
17.37 I.M. Trancoso, B.S. Atal: Efficient procedures for finding the optimum innovation in stochastic coders, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1986) pp. 2375–2378
17.38 G. Davidson, A. Gersho: Complexity reduction methods for vector excitaiton coding, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1986) pp. 3055–3058
17.39 D. Lin: Speech coding using pseudo-stochastic block codes, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1987) pp. 1354–1357
17.40 I.A. Gerson, M.A. Jasiuk: Vector sum excited linear prediction (VSELP), Proc. 1989 IEEE Workshop Speech Coding for Telecommunications (1989) pp. 66–68
17.41 P. Kroon, B.S. Atal: Pitch predictors with high temporal resolution, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1990) pp. 661–664
17.42 J.S. Marques, I.M. Trancoso, J.M. Tribolet, L.B. Almeida: Improved pitch prediction with fractional delays in CELP coding, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1990) pp. 665–668
17.43 J.H. Chen, M.S. Rauchwerk: An 8 kb/s low-delay CELP speech coder, Proc. IEEE Global Commun. Conf. (1991) pp. 1894–1898
17.44 S. Miki, K. Mano, H. Ohmuro, T. Moriya: Pitch synchronous innovation CELP (PSI-CELP), Proc. 1993 Eurospeech Conf. (1993) pp. 261–264
17.45 T. Moriya: Two-channel conjugate vector quantization for noisy channel speech coding, IEEE J. Sel. Areas Commun. **10**(5), 866–874 (1992)
17.46 ITU-T: G.729: coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction (CS-ACELP) (1996)
17.47 J.-P. Adoul, C. Lamblin: A comparison of some algebraic structures for CELP coding of speech, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1987) pp. 1953–1956
17.48 R.A. Salami: Binary code excited linear prediction (BCELP): a new approach to CELP coding of speech without the codebooks, IEEE Electron. Lett. **25**(6), 401–403 (1989)
17.49 A. Le Guyader, D. Massaloux, J.P. Petit: Robust and fast code-excited linear predictive coding of speech signals, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1989) pp. 120–122
17.50 J.-P. Adoul, C. Lamblin, A. Leguyader: Baseband speech coding at 2400 BPS using spherical vector quantization, IEEE Int. Conf. Commun. (1984) pp. 1.12.1–1.12.4
17.51 C. Laflamme, J.-P. Adoul, S. Morisette: A real time 4.8 Kbits/sec CELP on a single DSP chip (TMS320C25), IEEE Workshop Speech Coding for Telecommun. (1989) pp. 35–36

17.52 R.A. Salami, D.G. Appleby: A new approach to low bit rate speech coding with low complexity using binary pulse excitation (BPE), IEEE Workshop Speech Coding for Telecommun. (1989) pp. 63–65

17.53 C. Laflamme, J.-P. Adoul, R. Salami, S. Morisette, P. Mabilleau: 16 kbps wideband speech coding technique based on algebraic CELP, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1991) pp. 13–16

17.54 R. Salami, C. Laflamme, J.-P. Adoul, D. Massaloux: A toll quality 8 kb/s speech codec for the personal communications system (PCS), IEEE Trans. Vehicular Technol. **43**(3), 808–816 (1994)

17.55 K. Järvinen, J. Vaino, P. Kapanen, T. Honkanen, P. Haavisto, R. Salami, C. Laflamme, J.-P. Adoul: GSM enhanced full rate speech codec, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1997) pp. 771–774

17.56 ETSI EN 300 726 V8.0.1: Digital cellular telecommunications systems (Phase 2+); enhanced full rate (EFR) speech transcoding; (GSM 06.60 version 8.0.1 Release 1999) (2000)

17.57 3GPP TS 26.190 V6.1.1: 3rd generation partnership project; technical specification group services and system aspects; speech codec speech processing functions; adaptive multi-rate – wideband (AMR-WB) speech codec; transcoding functions (release 6) (2005)

17.58 R. Salami, C. Laflamme, J.P. Adoul, A. Kataoka, S. Hayashi, T. Moriya, C. Lamblin, D. Massaloux, S. Proust, P. Kroon, Y. Shoham: Design and description of CS-ACELP: a toll quality 8 kb/s speech coder, IEEE Trans. Speech Audio Process. **6**(2), 116–130 (1998)

17.59 ITU-T: G.729 Annex A: reduced complexity 8 kbit/s CS-ACELP speech codec (1996)

17.60 R. Salami, C. Laflamme, B. Bessette, J.P. Adoul: ITU-T G.729 Annex A: reduced complexity 8 kb/s CS-ACELP Codec for simultaneous vocie and data, IEEE Commun. Mag. **35**, 56–63 (1997)

17.61 ITU-T: G.722.2: wideband coding of speech at around 16 kbit/s using adaptive multi-rate wideband (AMR-WB) (2002)

17.62 3GPP TS 26.090 V6.0.0: 3rd generation partnership project; technical specification group services and system aspects; mandatory speech codec speech processing functions; adaptive multi-rate (AMR) speech codec; transcoding functions (release 6) (2004)

17.63 ANSI/TIA/EIA-136-410-99: TDMA cellular PCS – radio interface enhanced full-rate voice codec (ANSI/TIA/EIA-136-410-99) (R2003) (1999)

17.64 3GPP2 C.S0030-0 V1.0: Selectable mode vocoder service option for wideband spread spectrum communication systems, (June 15 2001)

17.65 ISO/IEC 14496-3 FCD, ISO/JTC 1/SC 29 N2203CELP: Information technology – coding of audiovisual objects, Part 3: audio, subpart 3: CELP, (May 13 1998)

17.66 A. Kataoka, T. Moriya, S. Hayashi: An 8-kbit/s speech coder based on conjugate structure CELP, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1993) pp. II–592–II–595

17.67 T. Moriya, H. Suda: An 8 kbit/s transform coder for noisy channels, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1989) pp. 196–199

17.68 A. Kataoka, T. Moriya, S. Hayashi: An 8-kb/s conjugate structure CELP (CS-CELP) speech coder, IEEE Trans. Speech Audio Process. **4**(6), 401–411 (1996)

17.69 W.B. Kleijn, P. Kroon, L. Cellario, D. Sereno: A 5.85 kb/s CELP algorithm for cellular applications, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1993) pp. II–596–II–599

17.70 W.B. Kleijn, R.P. Ramachandran, P. Kroon: Interpolation of the pitch-predictor parameters in analysis-by-synthesis speech coders, IEEE Trans. Speech Audio Process. **2**(1), 42–53 (1994)

17.71 Y. Gao, A. Benyassine, J. Thyssen, H. Su, E. Shlomot: A speech coding paradigm, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (2001) pp. 689–692

17.72 J.-H. Chen: Novel codec structures for noise feedback coding of speech, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (2006) pp. I.681–I.684

17.73 J.D. Makhoul, M. Berouti: Adaptive noise spectral shaping and entropy coding in predictive coding of speech, IEEE Trans. Acoust. Speech Signal Process. **27**, 63–73 (1979)

17.74 J. Thyssen, J.-H. Chen: Efficient VQ techniques and general noise shaping for noise feedback coding, Proc. Interspeech 2006 ICSLP (2006) pp. 221–224

17.75 PacketCable 2.0 codec and media specification, PKT-SP-CODEC-MEDIA-I02-061013 (2006)

17.76 ITU-T: G.729.1: G.729 based embedded variable bitrate coder: An 8-32 kbit/s scalable wideband coder bitstrean interoperable with G.729 (2006)

17.77 ITU-T: G.726: 40, 32, 24, 16 kbit/s adaptive differential pulse code modulation (ADPCM) (1990)

17.78 ITU-T: G.722: 7 kHz audio-coding within 64 kbit/s (1988)

16. Low-Bit-Rate Speech Coding

A. V. McCree

Low-bit-rate speech coding, at rates below 4 kb/s, is needed for both communication and voice storage applications. At such low rates, full encoding of the speech waveform is not possible; therefore, low-rate coders rely instead on parametric models to represent only the most perceptually relevant aspects of speech. While there are a number of different approaches for this modeling, all can be related to the basic linear model of speech production, where an excitation signal drives a vocal-tract filter.

The basic properties of the speech signal and of human speech perception can explain the principles of parametric speech coding as applied in early vocoders. Current speech modeling approaches, such as mixed excitation linear prediction, sinusoidal coding, and waveform interpolation, use more-sophisticated versions of these same concepts. Modern techniques for encoding the model parameters, in particular using the theory of vector quantization, allow the encoding of the model information with very few bits per speech frame.

Successful standardization of low-rate coders has enabled their widespread use for both military and satellite communications, at rates from 4 kb/s all the way down to 600 b/s. However, the goal of toll-quality low-rate coding continues to provide a research challenge.

16.1 Speech Coding

As digital computers and communication systems continue to spread through our modern society, the use of digitized speech signals is increasingly common. The large number of bits required for accurate reproduction of the speech waveform makes many of these systems complex and expensive, so more-efficient encoding of speech signals is desirable. For example, limited radio bandwidth is a major constraint in the design of public mobile telephone systems, and the speech data rate directly influences the bandwidth requirement. In military tactical communications, a system with a lower speech data rate can use less transmitter power to make detection more difficult, or it can allow higher signal-to-noise ratios to improve performance in a hostile jamming environment. Also, computer storage of speech, such as in voice mail or voice response systems, becomes cheaper if the number of bits required for speech storage can be reduced. These are just some of the applications that can benefit from the development of algorithms that significantly reduce the speech data rate.

At bit rates above 4 kb/s, speech-specific waveform coders based on code excited linear prediction (CELP) [16.1] can produce good-quality speech. But at lower rates, it becomes very difficult to encode all of the information in the speech signal. Therefore, most low-rate coders model only the key perceptual features of speech, rather than the entire waveform. Since this is typically done by encoding the parameters of a linear model for speech, these are called parametric speech coders.

In Sect. 16.2, this presentation of low-rate speech coding begins with a review of the basics of human speech production and perception, and then introduces the linear model of speech as the basis for parametric speech coding. Section 16.3 then presents modern low-rate speech coding models using mixed excitation linear prediction, sinusoidal coding, and waveform interpolation. After a discussion of techniques for quantizing the model parameters in Sect. 16.4, complete low-rate coder designs that have been standardized for use in a range of applications are presented in Sect. 16.5, followed by conclusions and a summary in Sect. 16.6.

16.2 Fundamentals: Parametric Modeling of Speech Signals

For parametric coding, it is important to understand the properties of speech signals and which of their characteristics need to be preserved to provide high quality to a human listener. Therefore, in this section, we review human speech production and perception and then introduce the classical vocoder algorithms.

16.2.1 Speech Production

Speech is a sequence of sounds generated by the human vocal system [16.2, 3]. The acoustic energy necessary to produce speech is generated by exhaling air from the lungs. This air stream is used to produce sound in two different ways: by vibrating the vocal cords or by forcing air turbulence. If the vocal cords are used, the speech is referred to as voiced speech; otherwise, the speech is called unvoiced. In voiced speech, the opening and closing of the vocal cords at the glottis produces quasiperiodic puffs of air called glottal pulses, which excite the acoustic tubes of the vocal and nasal tracts. The average spacing between glottal pulses is called the pitch period. The frequency content of the resulting acoustic wave propagating from the mouth depends on the spectrum of the glottal pulses and the configuration of the vocal tract. Typically, glottal pulses are roughly triangular in shape with a sharp discontinuity at the closure of the glottis, as shown in Fig. 16.1. As this stylized example illustrates, glottal pulses have most of their energy concentrated at lower frequencies.

Unvoiced sounds result from turbulent noise exciting the vocal and nasal tracts. The turbulence can come from simply breathing air out of the lungs in a process called aspiration. This is the excitation source for whispered speech and for the /h/ sound. Turbulence can also be

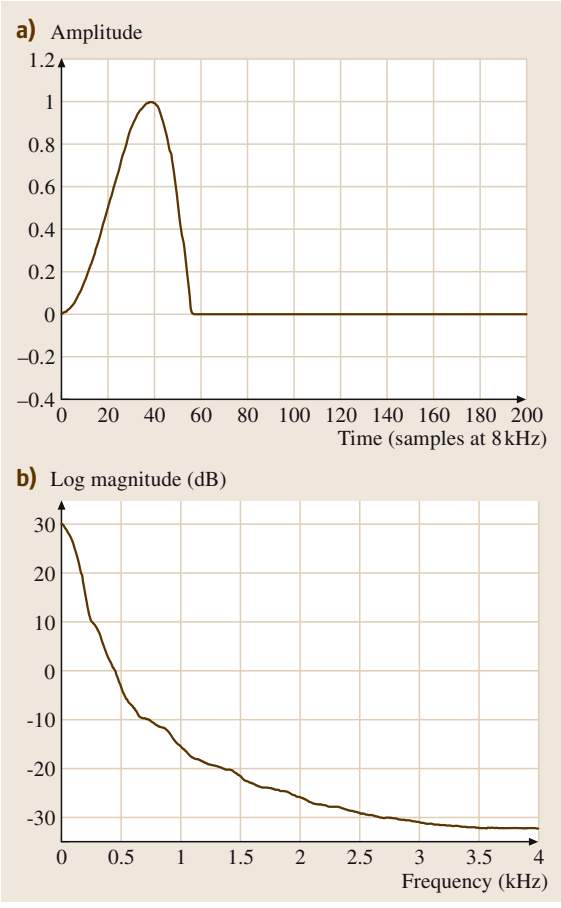


Fig. 16.1 (**a**) Stylized glottal pulse and (**b**) corresponding Fourier transform (after *Rabiner* and *Schafer* [16.2])

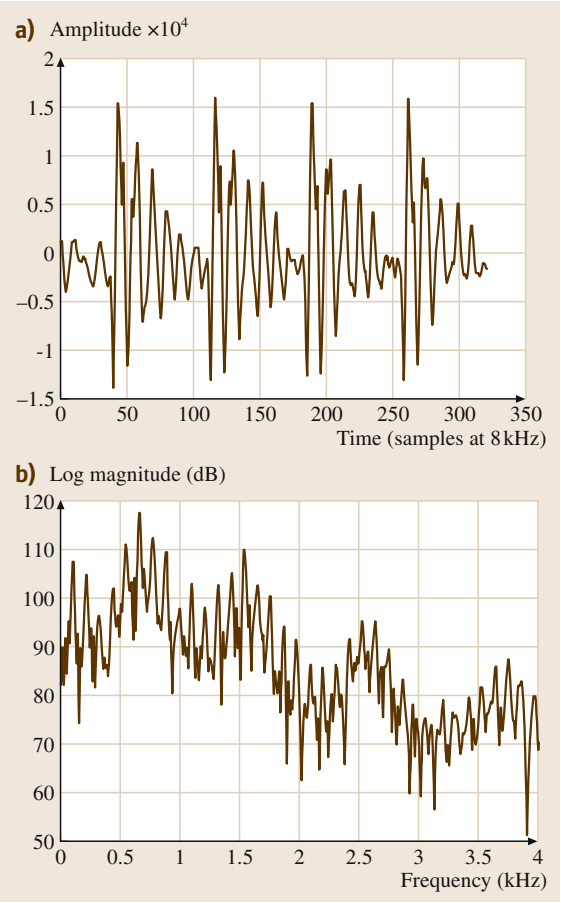


Fig. 16.2a,b Sustained vowel /æ/: **(a)** waveform and **(b)** Fourier spectrum

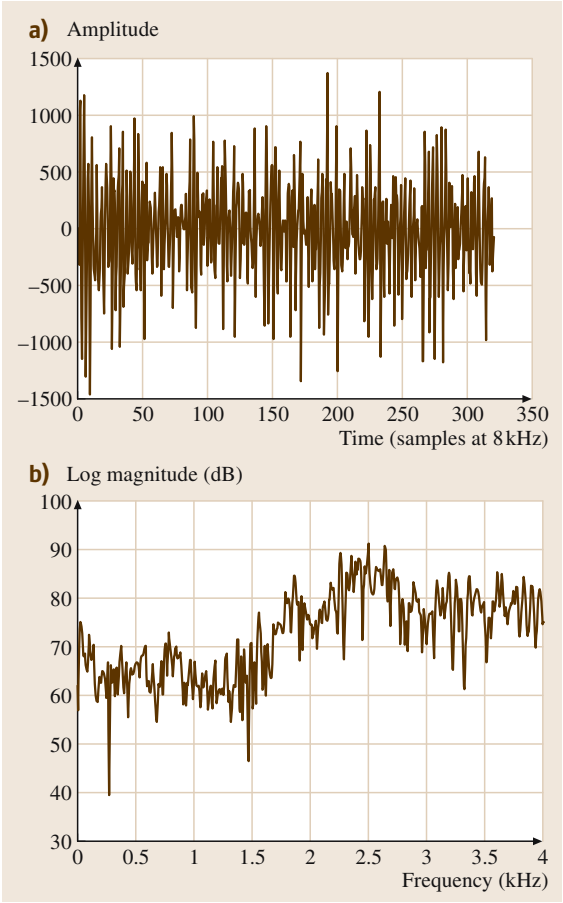


Fig. 16.3a,b Sustained fricative /sh/: **(a)** waveform and **(b)** Fourier spectrum

generated by forcing the air through a constriction in the vocal tract formed with the tongue or lips. This process, called frication, is used to make sounds such as /sh/.

Figures 16.2–16.4 show time waveforms and Fourier spectra for some typical speech sounds. Figure 16.2 shows the waveform and spectrum from a sustained vowel. Since this vowel was generated by periodic glottal pulses, the spectrum consists of harmonics of the pitch fundamental, in this case 110 Hz. The spectral peaks at 600, 1500, 2500, and 3600 Hz represent the formant frequencies in this example. The relatively high amplitudes of the first few harmonics are a result of the glottal pulse excitation. Figure 16.3 shows the waveform and spectrum of a sustained fricative. This sound is mainly high-frequency turbulent noise. An example of a plosive sound is shown in Fig. 16.4. Since this is an unvoiced plosive, the signal consists of silence during the vocal tract closure followed by a broadband noise burst.

The basic features of the speech production process can be captured in a simple time-varying linear model, as shown in Fig. 16.5. In this model, glottal pulses or random noise pass through a linear filter representing the vocal-tract frequency response. The parameters describing the excitation and vocal tract change as a function of time, tracking changes in the speech waveform. Key parameters include the voiced/unvoiced decision, pitch period, glottal pulse shape, vocal-tract filter coefficients, and power level. This model can directly mimic stationary sounds, whether voiced or unvoiced, by appropriate choice of excitation signal and vocal tract frequency response. Also, it can approximate nonstationary sounds if the model parameters are changed as rapidly as in natural speech.

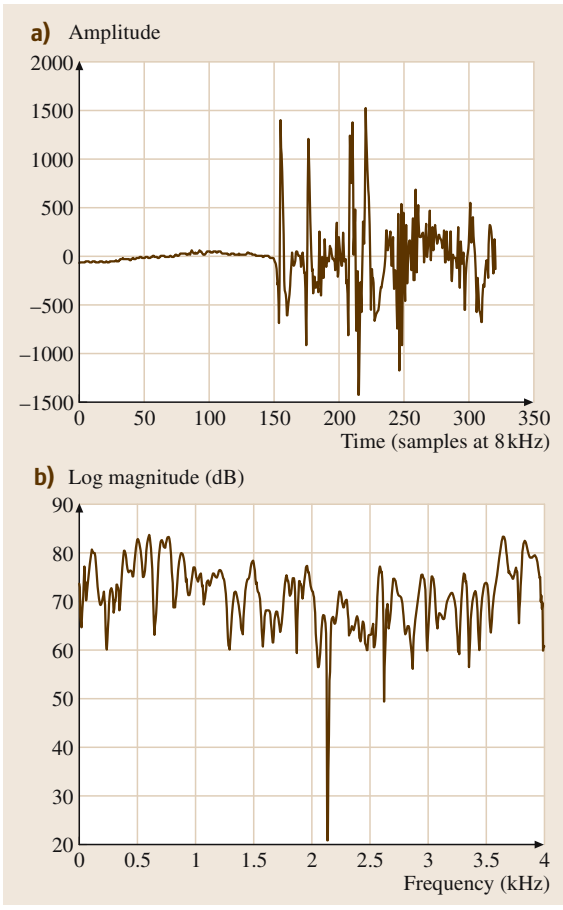


Fig. 16.4a,b Plosive /t/: **(a)** waveform and **(b)** Fourier spectrum

16.2.2 Human Speech Perception

Human perception of speech is determined by the capabilities of the human auditory system, which consists of the ear, the auditory nerve, and the brain [16.4]. The ear serves as a transducer, converting the acoustic input at the outer ear first to bone vibrations in the middle ear, then to fluid motion in the cochlea of the inner ear, and finally to electrical pulses generated by the inner hair cells in the cochlea. The location of maximum fluid vibration in the cochlea varies systematically with the input signal frequency, and this frequency response varies with the strength of the input signal. Also, the inner hair cells only detect motion in one direction. Thus, the ear acts like a very large bank of bandpass filters with dynamic range compression, and the output of each filter undergoes half-wave rectification. The half-wave-rectified bandpass-filtered outputs are transmitted across the auditory nerve to the lower levels of the brain, where specialized neurons can perform basic signal-processing operations. For example, there are neurons that respond to onsets and to decays, while other neurons may be able to estimate autocorrelations by comparing a signal to a delayed version of itself. The outputs of these neurons are then passed to higher levels of the brain for more-sophisticated processing. This results in the final analysis of the acoustic signal based on context and additional knowledge, such as classification of sound source and interpretation of pattern. In speech processing, this includes recognition of words as well as analysis of speaker identity and emotional state.

For a typical speech signal such as the vowel shown in Fig. 16.2, a human listener should be able to distin-

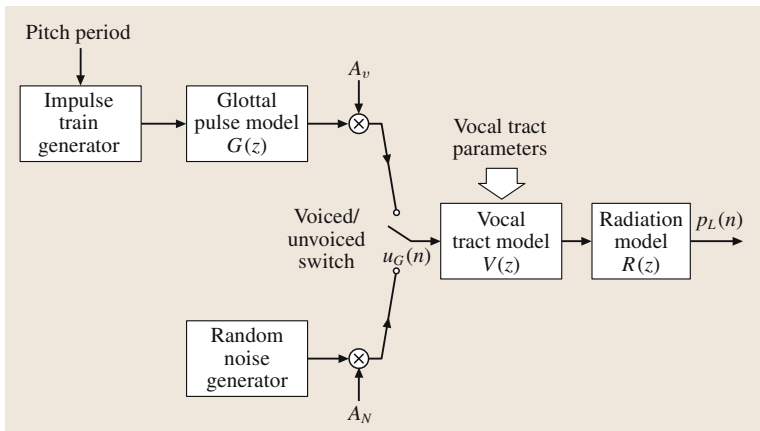


Fig. 16.5 Simple linear model of speech production (after *Rabiner* and *Schafer* [16.2])

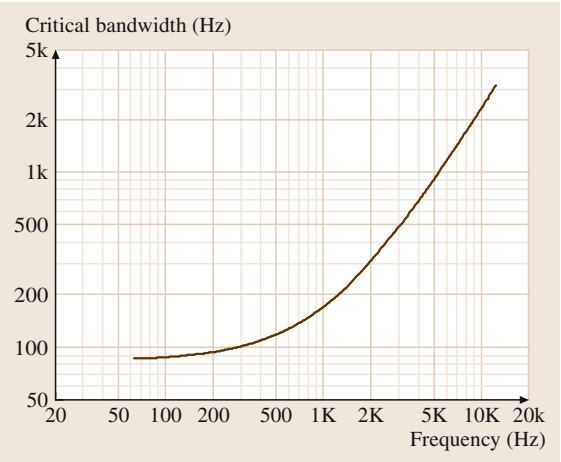


Fig. 16.6 Critical bands as a function of center frequency

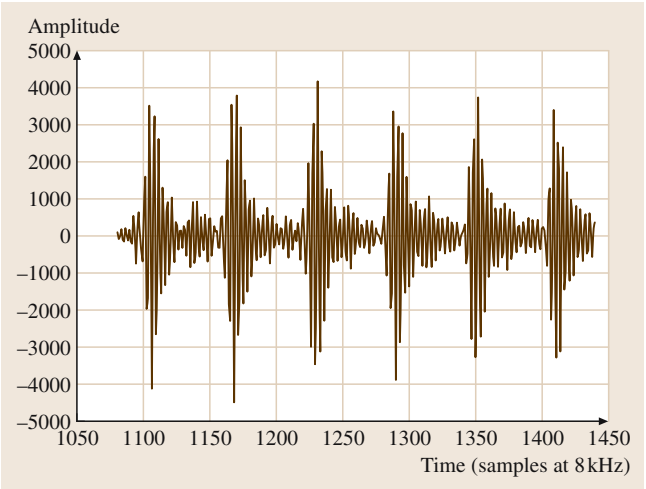


Fig. 16.7 Example bandpass filter output

guish a number of features. A comparison of the average power in each bandpass filter gives an overall estimate of the power spectrum of the speech signal. This spectral analysis will also yield the exact frequency of the lower pitch harmonics. However, human listeners cannot distinguish individual frequencies present in a 12 tone signal unless the tones are separated by more than a measure of bandwidth called the critical band [16.5], and these bandwidths increase with frequency as shown in Fig. 16.6. Therefore, for a typical male pitch frequency of 100 Hz, individual pitch harmonics will not be distinguished above about 500 Hz. Higher-frequency harmonics near the formant frequencies may also be detected by this spectral analysis if they are much stronger than their immediate neighbors.

At higher frequencies, individual pitch harmonics cannot be resolved since each bandpass filter contains multiple pitch harmonics. However, there is a great deal of information in the time variation of the higher-frequency bandpass filter outputs. An example of a bandpass filter output centered at 2500 Hz from a sustained vowel is shown in Fig. 16.7. Notice that the individual pitch pulses can be seen in the rise and fall of this bandpass signal. It is reasonable to presume that the brain can detect pitch pulses as abrupt power transitions in the bandpass signals, and may also estimate pitch frequency from the average pulse spacing and voicing strength from the abruptness of the pulse onset. In fact, an experiment has shown that higher-frequency onset neurons in the cat respond with every pitch pulse in a sustained synthetic speech vowel [16.6].

Overall, the combination of spectral and bandpass waveform analysis allows the perception of many aspects of speech signals. For sustained or continuant sounds, the human listener should be able to estimate the spectral envelope, gain, harmonic content at lower frequencies, and pulse characteristics at higher frequencies. For noncontinuants, the human auditory system can detect rapid transitions in these characteristics, especially at high frequencies. These are the characteristics of the speech signal that we would expect low-rate coders to be able to reproduce.

16.2.3 Vocoders

Knowledge of speech production and speech perception can be combined to gain insight into efficient transmission and storage of speech signals. The ensemble of possible speech waveforms is fundamentally limited by the capabilities of the speech production process, and not all details of these waveforms are perceptually important. One particularly efficient method of speech compression is to analyze the parameters of the linear speech model, transmit these parameters across the communication channel, and synthesize a reproduction of the speech signal with a linear model. Speech coders based on this approach are called *vocoders* or *parametric* coders. If the parameters are updated 20 to 100 times per second, the synthesizer can reproduce much of the information of the speech signal using significantly fewer bits than direct waveform coding.

The first vocoder was developed by Homer *Dudley* in the 1930s [16.7]. In this method, the input speech is divided into 10 frequency channels, and the spectral envelope information representing the vocal tract is

transmitted as the average power in each channel. The synthesizer applies either a pulse train or random noise to an identical bank of bandpass filters. This approach, now called a channel vocoder, can produce speech that is intelligible but not of very high quality.

Because it uses only a fairly small number of frequency values to represent the spectral envelope, the channel vocoder cannot accurately reproduce the perceptually important formant structure of voiced speech. A better approach, called the formant vocoder [16.8, 9], transmits information about the formant frequencies directly and synthesizes speech with a small number of damped resonators or poles. The poles can be connected either in parallel or in cascade. Since they better model the spectral envelope of voiced speech, formant vocoders can produce speech of higher quality than channel vocoders. Unfortunately, the formants are difficult to estimate reliably in the analysis procedure, so formant vocoders may make annoying errors in tracking the formant frequencies. Formant synthesizers are more often used in text-to-speech applications where the formant frequencies can be generated by fixed rules.

The difficulty of formant tracking can be avoided if more poles are used to model the entire speech spectrum, without explicit identification of formant frequencies. In this way, the formants will be automatically captured when they dominate the spectrum, but more-subtle spectral features due to glottal pulse shape or nasal zeros can still be approximately reproduced. This principle is used in vocoders based on *linear predictive coding* (LPC), first introduced around 1970 [16.10, 11]. The term linear prediction is used because the all-pole coefficients are derived from the best predictor of the current input speech sample from a linear combination of previous samples:

$$\hat{s}(n) = \sum_{m=1}^{P} a_m s(n-m) \,, \tag{16.1}$$

where a_m are the set of P predictor coefficients. Minimum mean-square error estimation results in a highly efficient time-domain algorithm to generate these filter coefficients [16.2]. A block diagram of the LPC synthesizer is shown in Fig. 16.8. Either a periodic impulse train or white noise is used to excite an all-pole filter, and the output is scaled to match the level of the input speech. The voicing decision, pitch period, filter coefficients, and gain are updated for every block of input speech (called a speech frame) to track changes in the input speech.

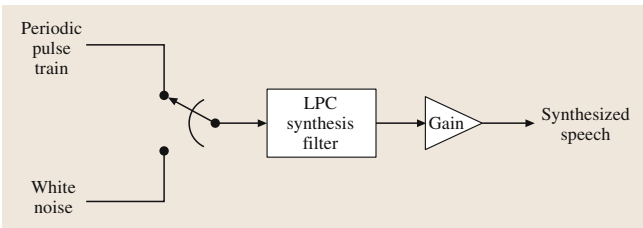


Fig. 16.8 LPC vocoder synthesizer

Since the basic LPC vocoder does not produce high-quality speech, there has been significant effort aimed at improving the standard model. One well-known problem with vocoder speech output is a strong buzzy quality. Formant synthesizers use a *mixed excitation* with simultaneous pulse and noise components to address this problem [16.8, 9], an idea which has also been used in channel vocoders [16.12] and LPC vocoders [16.13–15]. Most of these early mixed excitation systems used a low-pass-filtered pulse combined with high-pass-filtered noise, although a multiband mixture algorithm with separate voicing decisions in each of three bands was used in [16.12]. Other attempts at vocoder improvement included using more realistic excitation pulses [16.16] and pitch-synchronous LPC analysis [16.17]. *Atal* and *David* proposed encoding the Fourier coefficients of the LPC excitation signal [16.18]. Mathematically, this can be written as:

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) \exp(\mathrm{j}\omega_k n) \,, \tag{16.2}$$

where $x(n)$ is the excitation signal, N is the pitch period, k is a frequency index, $X(k)$ is a complex amplitude, and $\omega_k = \frac{2\pi k}{N}$ is the frequency of the k-th harmonic. Since $x(n)$ is real, this can equivalently be written as:

$$x(n) = \sum_{k=0}^{N/2} [A_\mathrm{c}(k) \cos(\omega_k n) + A_\mathrm{s}(k) \sin(\omega_k n)] \,, \tag{16.3}$$

where $A_\mathrm{c}(k)$ and $A_\mathrm{s}(k)$ are real amplitudes, or as:

$$x(n) = \sum_{k=0}^{N/2} A(k) \cos(\omega_k n + \phi_k) \,, \tag{16.4}$$

where $A(k)$ and ϕ_k represent the magnitudes and phases of the harmonics. These coefficients were analyzed with either a pitch-synchronous Fourier series analysis or a frame-based Fourier transform, and the excitation was synthesized as a sum of harmonic sine waves.

While these techniques provided some improvement, as of the early 1980s these enhanced LPC vocoders

were still unable to provide reliable communications in all environments. However, they did provide the background for newer approaches that have provided significant improvements in speech quality at low rates. The more-recent technical progress enabling this success has come in two areas: flexible parametric models and efficient quantization techniques. These are the subjects of the next two sections.

16.3 Flexible Parametric Models

Three more-recent speech models have been shown to provide significant performance improvements over the traditional LPC vocoder model: mixed excitation linear prediction, sinusoidal coding, and waveform interpolation. In this section, we review each of these approaches.

16.3.1 Mixed Excitation Linear Prediction (MELP)

The primary feature of the MELP model, shown in Fig. 16.9, is the use of mixed excitation for more-robust LPC synthesis [16.19]. Other key features are the use of aperiodic pulses, adaptive spectral enhancement to match formant waveforms, and a pulse dispersion filter to better match natural excitation waveform characteristics. Finally, MELP also includes representation of the Fourier magnitudes of the excitation signal.

Mixed Excitation

MELP generates an excitation signal with different mixtures of pulse and noise in each of a number (typically five) of frequency bands [16.20]. As shown in Fig. 16.9, the pulse train and noise sequence are each passed through time-varying spectral shaping filters and then added together to give a full-band excitation. For each frame, the frequency shaping filter coefficients are generated by a weighted sum of fixed bandpass filters. The pulse filter is calculated as the sum of each of the bandpass filters weighted by the voicing strength in that band. The noise filter is generated by a similar weighted sum, with weights set to keep the total pulse and noise power constant in each frequency band. These two frequency shaping filters combine to give a spectrally flat excitation signal with a staircase approximation to any desired noise spectrum. Since only two filters are required regardless of the number of frequency bands, this structure is more computationally efficient than using a bank of bandpass filters [16.19].

To make full use of this mixed excitation synthesizer, the desired mixture spectral shaping must be estimated accurately for each frame. In MELP, the relative pulse and noise power in each frequency band is determined by an estimate of the voicing strength at that frequency in the input speech. An algorithm to estimate these voicing strengths combines two methods of analysis of the bandpass-filtered input speech. First, the periodicity in each band is estimated using the strength of the normalized correlation coefficients around the pitch lag, where the correlation coefficient for a delay t is defined by

$$c(t) = \frac{\sum_{n=0}^{N-1} s(n)s(n+t)}{\sqrt{\sum_{n=0}^{N-1} s^2(n) \sum_{n=0}^{N-1} s^2(n+t)}} . \tag{16.5}$$

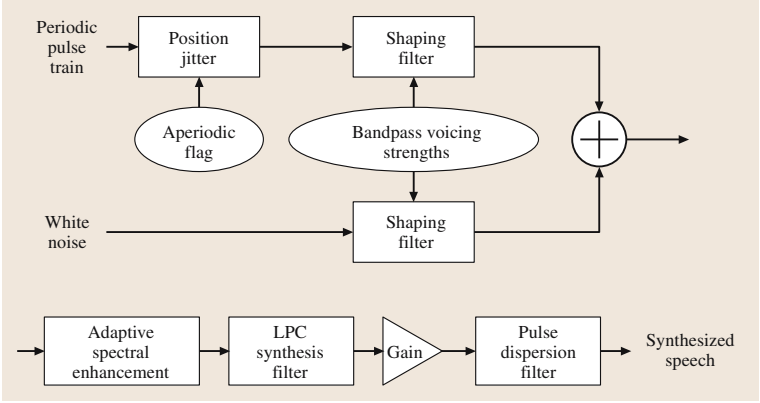


Fig. 16.9 MELP synthesizer

This technique works well for stationary speech, but the correlation values can be too low in regions of varying pitch. The problem is worst at higher frequencies, and results in a slightly whispered quality to the synthetic speech. The second method is similar to the time-domain analysis of a wideband spectrogram. The envelopes of the bandpass-filtered speech are generated by full-wave rectification and smoothing with a one-pole low-pass filter, while using a first-order notch filter to remove the direct-current (DC) term from the output. At higher frequencies, these envelopes can be seen to rise and fall with each pitch pulse, just as in the spectrogram display. Autocorrelation analysis of these bandpass filter envelopes yields an estimate of the amount of pitch periodicity. The overall voicing strength in each frequency band is chosen as the largest of the correlation of the bandpass filtered input speech and the correlation of the envelope of the bandpass-filtered speech.

Aperiodic Pulses

This mixed excitation can remove the buzzy quality from the speech output, but another distortion is sometimes apparent. This is the presence of short isolated tones in the synthesized speech, especially for female speakers. The tones can be eliminated by adding noise in the lower frequencies, but so much noise is required that the output speech sounds rough and noisy. A more-effective solution is to destroy the periodicity in the voiced excitation by varying each pitch period length with a pulse position jitter uniformly distributed up to ±25%. This allows the synthesizer to mimic the erratic glottal pulses that are often encountered in voicing transitions or in vocal fry [16.21]. This cannot be done for strongly voiced frames without introducing a hoarse quality, so a control algorithm is needed to determine when the jitter should be added.

Therefore, MELP adds a third voicing state to the voicing decision that is made at the transmitter [16.22]. The input speech is now classified as either voiced, jittery voiced, or unvoiced. In both voiced states, the synthesizer uses a mixed pulse/noise excitation, but in the jittery voiced state the synthesizer uses aperiodic pulses as shown in Fig. 16.9. This makes the problem of voicing detection easier, since strong voicing is defined by periodicity and is easily detected from the strength of the normalized correlation coefficient of the pitch search algorithm. Jittery voicing corresponds to erratic glottal pulses, so it can be detected by either marginal correlation or peakiness in the input speech. The peakiness p is defined by the ratio of the root-mean-square (RMS) power to the average value of the full-wave-rectified LPC residual signal [16.23]:

$$p = \frac{\sqrt{\frac{1}{N}\sum_{n=0}^{N-1} s^2(n)}}{\frac{1}{N}\sum_{n=0}^{N-1} |s(n)|} \,. \tag{16.6}$$

This peakiness detector will detect unvoiced plosives as well as jittery voicing, but this is not a problem since the use of randomly spaced pulses has previously been suggested to improve the synthesis for plosives [16.15].

Adaptive Spectral Enhancement

The third feature in the MELP model is adaptive spectral enhancement. This adaptive filter helps the bandpass-filtered synthetic speech to match natural speech waveforms in the formant regions. Typical formant resonances usually do not completely decay in the time between pitch pulses in either natural or synthetic speech, but the synthetic speech waveforms reach a lower valley between the peaks. This is probably caused by the inability of the poles in the LPC synthesis filter to reproduce the features of formant resonances in natural human speech. Here, there are two possible causes. First, the problem could be simply due to improper LPC pole bandwidth. The synthetic time signal may decay too quickly because the LPC pole has a weaker resonance than the true formant. Another possible explanation is that the true formant bandwidth may vary somewhat within the pitch period [16.8], and the synthetic speech cannot mimic this behavior. Informal experiments have shown that a time-varying LPC synthesis pole bandwidth can improve speech quality by modeling this effect, but it can be difficult to control [16.19].

The adaptive spectral enhancement filter provides a simpler solution to the problem of matching formant waveforms. This adaptive pole/zero filter was originally developed to reduce quantization noise between the formant frequencies in CELP coders [16.24]. The poles are generated by a bandwidth-expanded version of the LPC synthesis filter, where each z^{-1} term in the z-transform of the LPC filter is replaced by αz^{-1}, with $\alpha = 0.8$. Since this all-pole filter introduces a disturbing low-pass filtering effect by increasing the spectral tilt, a weaker all-zero filter calculated with α equal to 0.5 is used to decrease the tilt of the overall filter without reducing the formant enhancement. In addition, a simple first-order finite impulse response (FIR) filter is used to further reduce the low-pass muffling effect [16.25].

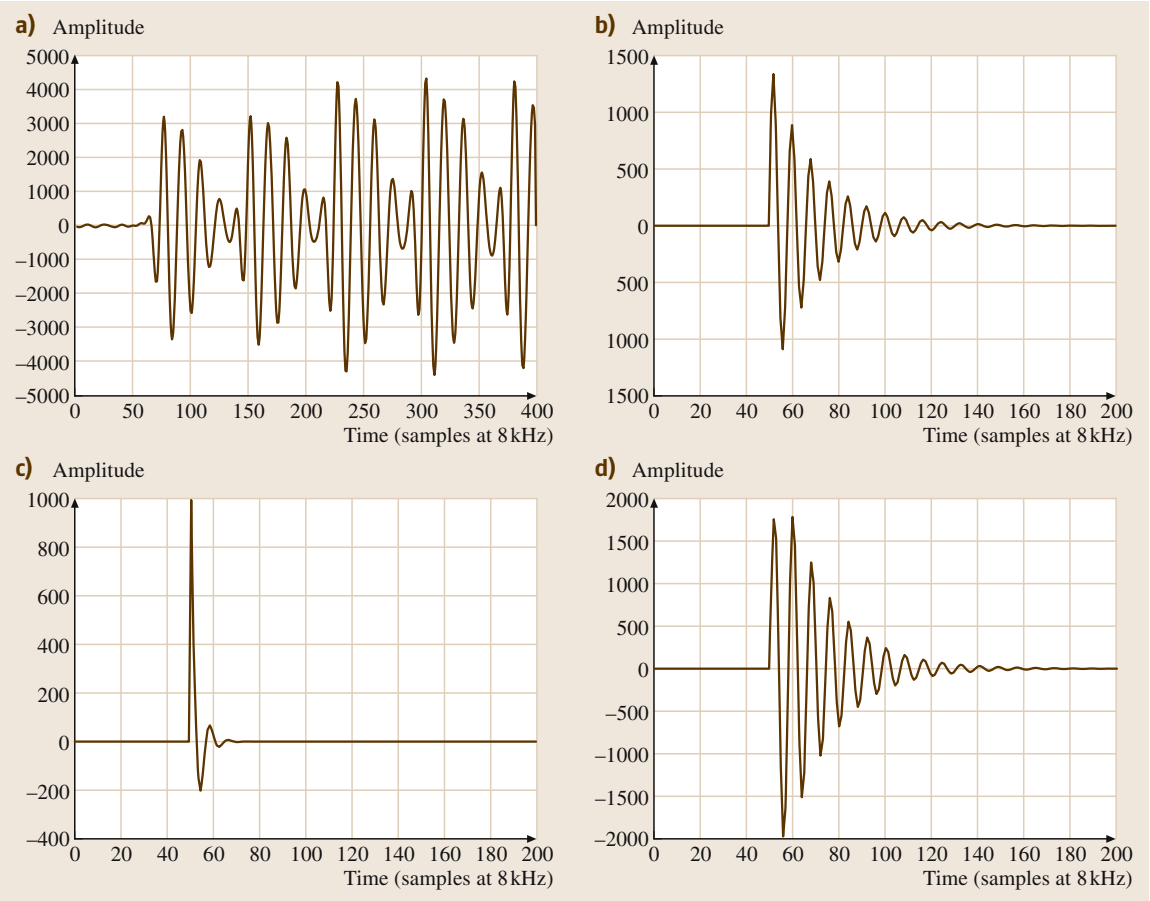


Fig. 16.10a–d Natural speech versus decaying resonance waveforms: **(a)** first formant of natural speech vowel, **(b)** synthetic exponentially decaying resonance, **(c)** pole/zero enhancement filter impulse response for this resonance, and **(d)** enhanced decaying resonance

In the MELP coder, reducing quantization noise is not a concern, but the time-domain properties of this filter produce an effect similar to pitch-synchronous pole bandwidth modulation. As shown in Fig. 16.10, a simple decaying resonance has a less abrupt time-domain attack when this enhancement filter is applied. This feature allows the speech output to better match the bandpass waveform properties of natural speech in formant regions, and increases the perceived quality of the synthetic speech.

Pulse Dispersion Filter

The pulse dispersion filter shown in Fig. 16.9 improves the match of bandpass-filtered synthetic and natural speech waveforms in frequency bands that do not contain a formant resonance. At these frequencies, the synthetic speech often decays to a very small value between the pitch pulses. This is also true for frequencies near the higher formants, since these resonances decay significantly between excitation points, especially for the longer pitch periods of male speakers. In these cases, the bandpass-filtered natural speech has a smaller peak-to-valley ratio than synthetic speech. In natural speech, the excitation may not all be concentrated at the point in time corresponding to the closure of the glottis [16.26]. This additional excitation prevents the natural bandpass envelope from falling as low as the synthetic version. This could be due to a secondary excitation peak from the opening of the glottis, aspiration noise resulting from incomplete glottal closure, or a small amount of acoustic background noise, which is visible in between the excitation peaks. In all these cases, there is a greater difference

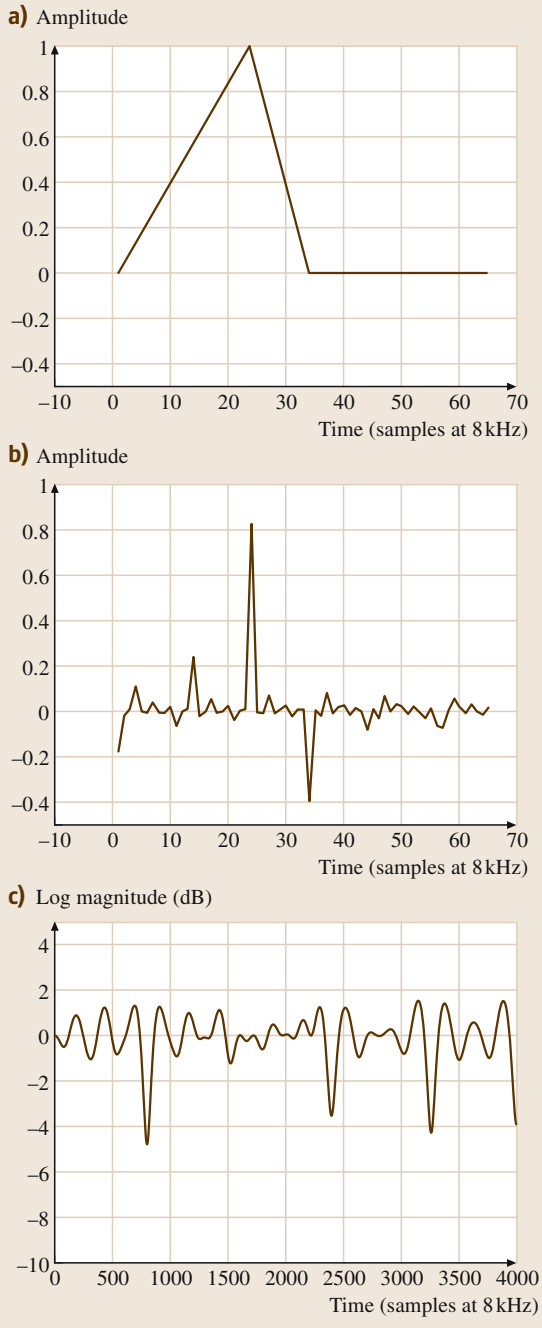


Fig. 16.11a–c Synthetic triangle pulse and FIR filter: **(a)** triangle waveform, **(b)** filter coefficients after spectral flattening with length 65 DFT, and **(c)** Fourier transform after spectral flattening

between the peak and valley levels of the bandpass-filtered waveform envelopes for the LPC speech than for natural human speech.

The pulse dispersion filter is a fixed FIR filter, based on a spectrally flattened synthetic glottal pulse, which introduces time-domain spread into the synthetic speech. In [16.19], a fixed triangle pulse [16.16, 27] based on a typical male pitch period is used, with the low-pass character removed from its frequency response. The filter coefficients are generated by taking a discrete Fourier transform (DFT) of the triangle pulse, setting the magnitudes to unity, and taking the inverse DFT. The dispersion filter is applied to the entire synthetic speech signal to avoid introducing delay into the excitation signal prior to synthesis. Figure 16.11 shows some properties of this triangle pulse and the resulting dispersion filter. The pulse has considerable time-domain spread, as well as some fine detail in its Fourier magnitude spectrum. Using this dispersion filter decreases the synthetic bandpass waveform peakiness in frequencies away from the formants and results in more natural sounding LPC speech output.

Fourier Series Modeling

The final feature of the MELP model is a Fourier series expansion of the voiced excitation signal. Implementing a Fourier series expansion in MELP synthesis is straightforward, using the pitch-synchronous Fourier series synthesis of (16.4). Instead of using a simple time-domain digital impulse, each pitch pulse of the mixed excitation signal shown in Fig. 16.9 is generated by an inverse DFT of exactly one period in length. If the magnitude values of this DFT are all equal to one and the phases are all set to zero, this is simply an alternate representation for a digital impulse. As in [16.18], these magnitudes can be estimated using either a pitch-synchronous Fourier series analysis or a DFT with a longer window.

MELP Model Improvements

A number of refinements to the basic MELP model have been demonstrated. Replacement of the frame-based autocorrelation pitch algorithm with a subframe-based approach that allows time-varying pitch within an analysis frame was shown to improve the reliability of pitch estimation in [16.28]. The use of a sliding pitch window, a specially designed plosive speech analysis and synthesis algorithm, and an excitation magnitude post-filter combined to improve performance in [16.29]. A wideband MELP coder, extended to 8 kHz

bandwidth, provided improved quality in [16.30]. An intelligibility enhancement preprocessor that emphasizes the perceptually important formant transitions by combining adaptive spectral enhancement with variable-rate timescale modification improved intelligibility in [16.31]. Finally, a number of enhancements, including subframe pitch estimation, pitch-synchronous LPC and Fourier series analysis, and oversampled Fourier synthesis for fractional pitch values, showed significant performance improvement for female speakers [16.32].

16.3.2 Sinusoidal Coding

Another successful approach to low-rate speech coding is based on modeling speech as a sum of sine waves [16.33,34]. Initial encouraging work in this direction [16.35,36] led to the development of two successful speech coding techniques: the sinusoidal transform coder (STC) [16.37] and the multiband excited (MBE) vocoder [16.38].

The Sinusoidal Model of Speech

Many signals can be locally modeled as a sum of sine waves:

$$s(n) = \sum_{l=0}^{L} A(l)\cos\left(\omega_l n + \phi_l\right) , \tag{16.7}$$

where $A(l)$, ω_l, and ϕ_l are the magnitude, frequency, and phase of the l-th sine wave. Note that, while this equation is similar to (16.4), in this case we are not constraining the frequencies to be harmonically related, and we are analyzing the speech signal itself rather than the excitation signal. While it is clear that such a model is appropriate for quasiperiodic voiced speech, it can be shown that this representation can also provide a perceptually sufficient representation of unvoiced speech as long as the sinusoids are closely spaced in frequency [16.37]. The principle of sinusoidal speech coding is to estimate and transmit the sine wave parameters for each speech frame, and then to synthesize speech using these parameters.

Sinusoidal Parameter Estimation

The general problem of estimating the time-varying sine wave parameters such that the reconstructed speech signal is as close as possible to the original speech is difficult to solve analytically. However, if a speech signal is locally periodic, then minimizing the squared estimation error over a window spanning a number of consecutive pitch periods produces a straightforward solution: the optimal estimates of the complex amplitudes are given by the DFT values at the harmonics of the pitch fundamental frequency [16.33]. Also, since the nonharmonic DFT coefficients will be zero in this case, the sine wave frequencies correspond to the peaks of this spectral estimate. Based on these insights, a more-general sinusoidal analysis method has been developed and shown to be very effective in STC [16.37]:

- window the input speech signal, for example with a Hamming window with a duration of approximately 2–3 pitch periods
- compute the DFT
- find the sine wave frequencies as the locations of the peaks of the DFT magnitude
- estimate the magnitude and phase of each sinusoid based on the corresponding complex DFT coefficients.

An alternative approach, used in the MBE vocoder, is to assume that the speech signal is locally periodic, but to model the DFT distortion due to windowing explicitly as a frequency-domain convolution of the desired sine wave coefficients with the DFT of the window function [16.38]. Then the sinusoid frequencies are the harmonics of the pitch fundamental, and the minimum squared error estimates of the amplitudes and phases come from the speech spectrum at frequencies around the harmonic along with the spectrum of the window function.

Finally, the sine wave parameters can be estimated using analysis by synthesis, either in the frequency or time domain [16.39–41].

Synthesis

At first glance, it would seem that sinusoidal synthesis using (16.7) is straightforward. However, each frame of speech has different sine wave parameters, so the synthesizer must perform smoothing to avoid discontinuities at the frame boundaries.

One form of parameter interpolation involves matching sinusoids from one frame to the next and then continuously interpolating their amplitudes, phases, and frequencies [16.37]. For well-behaved signals this matching process is not difficult, but a general solution typically involves an additional algorithm for birth and death of sine wave tracks to handle difficult cases. Phase interpolation also presents difficulties, since the phase values are only relative within a period, and the number of periods from one frame to the next must also be estimated. An elegant solution is to assume that the phase trajectory can be modeled with a cubic polynomial as

a function of time; this is the simplest solution meeting endpoint constraints for both frequency and phase.

A less-complex synthesis technique is the overlap/add method. Each frame is synthesized once with the constant parameters from the previous frame, and again with the new parameter set. The two synthesized signals are appropriately weighted by windowing functions and then summed together to produce the final synthesis output. These windows are designed such that the previous parameter synthesis tapers down to zero, the new parameter synthesis tapers up from zero, and the sum of the windows is one for every sample. Despite performing synthesis more than once for at least some of the speech samples, overlap/add synthesis can reduce complexity since there is no need to compute interpolated values of the parameters for every sample. In addition, it is readily amenable to the use of efficient fast Fourier transform methods.

Low-Rate Coding Constraints

The general sinusoidal model has been used successfully in many applications. Unfortunately, for low-rate speech coding, the complete set of sine wave amplitudes, phases, and frequencies typically contains too much information for efficient quantization. Therefore, the linear speech model is used to impose constraints on the sine wave parameters. First, the voiced speech frequencies are assumed to be harmonics of the fundamental frequency. This eliminates the need to estimate and transmit the exact frequency of each sinusoid, and incorporating this constraint into the sine wave analysis procedure naturally leads to integrated frequency-domain pitch and voicing estimation procedures. In both STC and MBE, pitch estimation is performed by measuring the optimal harmonic sinusoidal fit over a range of possible fundamental frequencies, and then selecting the pitch value as the one providing the best performance by a distance metric.

The linear speech model also necessitates the explicit modeling of unvoiced speech. Voicing decisions can be made based on the fit of the harmonic sinusoidal model to the original speech, since voiced speech frames should produce a better model fit. Typically better performance is obtained with a *soft* voicing decision, where partial voicing is allowed with both voiced and unvoiced components. As in earlier mixed excitation work, this is done either using a cutoff frequency, where frequencies below the cutoff are voiced and those above are unvoiced [16.37], or with separate decisions for a number of frequency bands [16.38]. The unvoiced component can be synthesized either with sinusoidal synthesis with randomized phases, or with white-noise generation.

Since encoding of the phases can require a significant number of bits, low-rate sinusoidal coders use a parametric model for the phase as well. Since the voiced excitation signal should look like a pulse, it can be assumed to be zero phase with a flat magnitude. In this case, the overall magnitude and phase responses will come from the vocal-tract filter.

In addition, the vocal tract can be modeled parametrically. In particular, LPC coefficients can be fitted to the sine wave amplitudes. This can be done with a frequency-domain algorithm, which can also incorporate a mel-scale frequency warping to improve perceptual modeling [16.42, 43], or with a time-domain LPC analysis [16.44]. The resulting model will have a minimum phase output corresponding to this all-pole filter. Alternatively, the sinusoidal amplitudes can be quantized using nonparametric techniques such adaptive transform coding [16.38] or direct vector quantization [16.45–47].

Spectral Post-filtering

Since the synthetic speech from sinusoidal coders sometimes exhibits a muffling effect, a postfilter can be applied to deepen the spectral valleys between formant frequencies [16.33]. Motivated by the time-domain LPC post-filter of [16.24], a frequency-domain post-filter design method has been developed [16.43]. For each frequency, these post-filter weights correspond to the unity-gain spectrally flattened sine-wave amplitudes raised to a power of less than one for spectral compression. This post-filtering approach is used in both STC and MBE coders to improve speech quality.

16.3.3 Waveform Interpolation

For voiced speech, it is natural to think of the excitation signal as a sequence of pitch periods or cycles, with the waveform for each cycle representing the ideal excitation for the vocal-tract filter. From one pitch period to the next, these excitation waveforms will be similar but not identical, since the speech content is gradually changing. Therefore, we can extract these pitch cycle excitation waveforms at a slower rate, quantize them for transmission, and reconstruct the missing waveforms at the receiver. This is the principle behind another successful low-rate speech technique called *waveform interpolation* (WI) [16.48].

Analysis

The fundamental operation of WI is the extraction of the excitation pitch cycle waveform, called the characteristic waveform [16.49]. Each waveform is extracted as one period of the signal, so minimizing edge effects is important for accurate representation. Therefore, this analysis is performed on the LPC residual signal; in addition, some local adjustment of the window position is allowed to minimize the signal power near the boundaries.

To ensure that the characteristic waveforms evolve smoothly in time, an alignment process is performed, in which each extracted waveform is circularly shifted to maximize its correlation with the previous one. The result of this alignment process is a sequence of time-aligned characteristic waveforms, each with a similar shape.

Synthesis

Synthesis in WI coders is typically done using harmonic sine wave synthesis of the excitation signal followed by LPC synthesis. The fundamental frequency and the complex sine wave coefficients are interpolated for every sample, while the LPC coefficients are interpolated on a subframe basis. However, it is sometimes more convenient to perform the LPC synthesis filtering in the frequency domain prior to the sine wave synthesis [16.48].

Note that, since the alignment phase (the circular time shift of each characteristic waveform) is not preserved, the precise time offset from the beginning to the end of a frame of synthesized speech is not explicitly controlled and will depend upon the interpolated pitch contour for the frame. Therefore, WI coders are typically not time-synchronized with the input speech signal; instead they exhibit a slow time drifting which is not perceptually relevant.

Low-Rate WI Speech Models

At low bit rates, WI coders also exploit the linear speech model. Besides the concepts of pitch and vocal-tract filtering mentioned in the basic WI formulation, additional bit savings can be introduced by using a soft voicing concept and a parametric model for system phase.

In WI, voicing is represented by the decomposition of the characteristic waveform into two additive components: the slowly evolving waveform (SEW) and the rapidly evolving waveform (REW) [16.50]. The underlying idea is that the quasiperiodic voiced component changes slowly over time, while the unvoiced noisy component changes quickly. Analysis of the SEW/REW decomposition is done by low-pass and high-pass filtering the aligned characteristic waveforms over time, typically with a cutoff frequency of around 20 Hz. The SEW can then be carefully quantized at an update rate of 40 Hz, while the REW can be replaced by a noise signal with similar spectral characteristics (typically by randomizing the phase). Since the full SEW waveform contains a great deal of information, at low rates it is typically quantized by coding only the lower harmonic amplitudes, setting the other amplitudes to unity, and setting the phases to zero so that the LPC filter will provide an overall minimum phase response. For the REW, only a crude amplitude envelope is necessary, but it should be updated quite frequently.

High-Rate Improvements

More recent work has improved the performance of WI modeling at higher bit rates. A WI method with asymptotically perfect reconstruction properties was presented in [16.51], enabling very high quality output from the WI analysis/synthesis system. Similarly, a scalable WI coder using pitch-synchronous wavelets allows a wide range of bit rates to be utilized [16.52]. In this coder, higher bit rates achieve better speech quality by encoding higher resolutions of the wavelet decomposition.

16.3.4 Comparison and Contrast of Modeling Approaches

All of the low-rate coding models described can be viewed as representing the linear speech model of Fig. 16.5, where the parameters for transmission for each frame of speech include the pitch period (or frequency), gain, voicing information, and vocal-tract filter. Common elements of most MELP, sinusoidal, and WI coders are:

- vocal-tract filter modeling with LPC
- pitch extraction algorithm for excitation analysis
- incorporation of excitation Fourier amplitude spectrum
- zero-phase excitation
- soft voicing decision, allowing mixed excitation synthesis
- spectral enhancement filtering to sharpen formant frequencies

However, each coder has its own strengths since it is based on a different perspective of the speech signal. The MELP model extends the traditional vocoder view of the excitation as a sequence of pulses, so it is able to exploit pitch pulse characteristics such as waveform shape. In contrast, sinusoidal modeling is based on a Fourier transform view of the speech signal, allowing straight-

forward frame-based processing. Finally, WI views the excitation signal as an evolving Fourier series expansion, allowing smooth analysis and synthesis of a signal with changing characteristics including pitch.

16.4 Efficient Quantization of Model Parameters

For low-rate speech coding, a flexible parametric model provides a necessary first step, but efficient quantization is critical to an overall coder design. In this section, we review two very important quantization topics: vector quantization, and LPC coefficient quantization.

16.4.1 Vector Quantization

The traditional approach to the quantization of speech parameters, such as harmonic amplitudes, is to encode each one separately. In contrast to this *scalar quantization* approach, significant performance improvements can be attained by encoding multiple values simultaneously, by searching a codebook of possible vectors. Since both encoder and decoder have copies of the same codebook, the only information to be transmitted is the index of the selected code vector. This process, called *vector quantization* (VQ) [16.53], has a long history in low-rate speech coding. The earliest applications, in pattern-matching vocoders of the 1950s, were based on tables designed for specific speech phonemes [16.54]. Starting around the same time, a more-rigorous information-theoretic approach was developing based on *Shannon*'s work on block source coding and rate-distortion theory [16.55].

Since the complexity of vector quantization can become prohibitive when the number of code vectors is large, specially structured codebooks are often used. Theoretically, such approaches will lose coding efficiency compared to a full VQ, but they can have significant advantages in codebook storage size and search complexity. In one common approach, called *split* VQ, the vector is divided into subvectors, and each of these is vector quantized. As the number of subvectors becomes large, split VQ becomes scalar quantization, so this approach provides a straightforward way to adjust coding performance versus complexity. Another approach, *multistage* VQ (MSVQ), maintains full dimensionality but reconstructs a quantized vector as the sum of code vectors from a number of different codebooks, each of smaller size. A simple MSVQ search algorithm, called sequential search, processes the codebooks one stage at at time. The input vector is quantized using the first codebook, then the error from the this first stage is quantized using the second codebook, and this process continues through all of the codebooks. However, much better performance is typically achieved with a joint search MSVQ, where the best joint set of code vectors over all stages is selected.

16.4.2 Exploiting Temporal Properties

Further coding efficiency can be attained by exploiting the slowly varying characteristics of speech over time. One simple approach uses *predictive* VQ, where the parameter values for a current frame are first predicted from past quantized values, and only the error between this predicted value and the true input is quantized. Typically a linear combination of past values is used, resulting in vector linear prediction equations similar to the traditional linear prediction of (16.1), but with prediction across frames rather than samples. These prediction coefficients can be designed offline based on training data; alternatively in *switched predictive VQ* the prediction coefficients can be adjusted and transmitted for each frame based on the speech characteristics or to optimize quantizer performance. By using simple models of speech evolution, predictive quantization can provide significant bit-rate reduction without introducing additional delay.

More-general coding schemes quantize a sequence of speech frames together. The advantage of joint encoding is that redundancies between neighboring frames can be fully exploited, but the disadvantage can be a significant increase in delay. A fixed block of parameter vectors, often called a *superframe*, can be encoded simultaneously. The spectral vectors can be quantized directly using *matrix quantization*, where each codebook entry corresponds to a sequence of vectors. Alternatively, adaptive interpolation can be used to transmit only a subset of frames with each block, replacing nontransmitted frames using interpolation. More flexibility can be introduced by replacing fixed blocks with variable-length segments. In particular, in a phonetic vocoder, each segment corresponds to a speech phoneme [16.56]. In this case, the only information transmitted is what was said (the phonetic content) and the way it was said (prosody).

16.4.3 LPC Filter Quantization

In a low-rate coder utilizing linear prediction, the LPC filter coefficients must be quantized. It is well known that these coefficients are not suitable for scalar quantization because they require a large number of bits to maintain an accurate representation and the quantized filter tends to be unstable. For this reason, early quantization work used alternative representations such as the reflection coefficients [16.2, 57]; however, almost all recent work on this topic uses the more-powerful line spectral frequency (LSF) representation [16.58].

Originally developed as a speech synthesis structure, the LSFs are generated by finding the roots of the two z-transform polynomials corresponding to the LPC coefficients with additional reflection coefficients of 1 or -1. LSFs have many useful properties for both quantization and synthesizer interpolation. They have a natural ordering, and a valid set of LSFs guarantees a stable LPC filter. In addition, the frequency representation allows quantization to take advantage of known properties of human perception. The higher frequencies can be quantized less accurately than lower frequencies, and LSFs corresponding to sharp LPC poles at the perceptually important speech formants can be selected for more-accurate representation.

In modern low-rate coders, spectral quantization is typically done with some form of vector quantization of the LSFs. A common performance measure is spectral distortion (SD) between unquantized and quantized LPC spectra, with a performance goal of average distortion of 1 dB while minimizing outlier frames [16.59]. However, there is evidence that a critical-band weighted version of SD is a more-accurate predictor of perceived distortion [16.60]. Typically, more than 20 bits are required per frame to achieve this performance, and the complexity of full VQ with 2^{20} codewords becomes too high for reasonable applications. Therefore, most low-rate coders use either split [16.59] or [16.61] multistage VQ. Finally, to implement VQ in the LSF domain, a weighted Euclidean LSF distance measure is needed. Commonly used functions weight LSFs in the vicinity of formants more strongly based on the LPC power spectrum [16.59], but the optimal LSF weighting to optimize SD performance has been derived [16.62]. This has also been modified to incorporate critical-band weighting [16.63].

16.5 Low-Rate Speech Coding Standards

In this section, we present a number of successful low-rate coder designs that have been standardized for communication applications. First, we describe low-rate coders that have been standardized for military communication applications: the US MIL-STD 3005 2.4 kb/s MELP and the North Atlantic Treaty Organization (NATO) STANAG 4591 MELPe coding suite at 2400, 1200, and 600 b/s. Then, we present an overview of speech coding standards for satellite telephony based on MBE. Finally, we discuss more-recent efforts by the International Telecommunications Union (ITU) to standardize a toll-quality low-rate coder at 4 kb/s.

Table 16.1 2.4 kb/s MELP coder bit allocation

Parameters	Voiced	Unvoiced
LSF	25	25
Fourier magnitudes	8	–
Gain (2 per frame)	8	8
Pitch and overall voicing	7	7
Bandpass voicing	4	–
Aperiodic flag	1	–
Error protection	–	13
Sync bit	1	1
Total bits/22.5 ms	54	54

16.5.1 MIL-STD 3005

In the early 1993, the US government Department of Defense (DoD) digital voice processing consortium initiated a three-year project to select a new 2.4 kb/s coder to replace the older LPC-10e standard [16.64] for secure communications. This selection process consisted of two parts: evaluation of six minimum performance requirements and overall composite rating by figure of merit [16.65]. Intelligibility, quality, speaker recognizability, communicability, and complexity were all measured as part of this selection process.

This project led to the selection in 1996 of a 2.4 kb/s MELP coder as the new MIL-STD 3005, as it had the highest figure of merit among the coders that passed all the requirements. This coder uses a 22.5 ms speech frame, with the bit allocation shown in Table 16.1 [16.66,67]. The spectral envelope is represented

by a 10th-order LPC, quantized using MSVQ of the LSFs with four stages and an M-best search algorithm [16.61]. Additional spectral modeling is achieved using the first 10 residual Fourier harmonic magnitudes, which are estimated from the spectral peaks of a frame-based DFT and quantized with an 8 bit VQ. The gain is estimated twice per frame, and quantized with a joint 8 bit coding scheme. The logarithm of the pitch period is quantized with a scalar quantizer spanning lags from 20 to 160 samples, with a reserved all-zero code for unvoiced frames. Of the five bandpass voicing decisions, the lowest one is represented by the overall voiced/unvoiced decision, and the remaining four are encoded with one bit each.

In terms of intelligibility, quality, speaker recognizability, and communicability, the performance of the 2.4 kb/s MELP standard was much better than the earlier 2.4 kb/s LPC-10e [16.68]. This coder also met the project goal of performance at least as good as the older 4.8 kb/s FS1016 CELP standard [16.69], at only half the bit rate. Interestingly, despite their differences in design, three other coders, based on STC, WI, and MBE, provided similar performance, with the first two of these also able to meet all six performance requirements.

16.5.2 The NATO STANAG 4591

These impressive results led the North Atlantic Treaty Organization (NATO) to undertake a similar standardization process in 1997, this time targeting the dual bit rates of 2.4 kb/s and 1.2 kb/s. Candidates from three NATO nations were evaluated using tests for speech quality, intelligibility, speaker recognition, and language dependency [16.70–72]. An enhanced MELP coder, called MELPe, was then selected as the 2.4/1.2 kb/s NATO standard STANAG 4591 in 2001 [16.73].

This 2.4 kb/s MELPe coder is based on MIL-STD 3005 MELP, using the same quantization techniques and bit allocation, and is therefore completely interoperable with it. However, performance has been improved, primarily by the addition of a robust noise suppression front-end based on the minimum mean-square error of the log spectral amplitudes [16.74]. This 2.4 kb/s MELPe coder was extended to also operate at 1.2 kb/s, using a superframe of three consecutive 22.5 ms frames [16.75]. This approach allows the 1.2 kb/s MELPe coder to perform nearly as well as the 2.4 kb/s version, at the price of additional coding delay.

In 1.2 kb/s MELPe, the pitch trajectory and voicing pattern are jointly vector quantized using 12 bits for each superframe. The LSF quantization depends upon the voicing pattern, but for the most challenging case of all voiced frames a forward/backward interpolation scheme is used. This algorithm has three components: the last frame is quantized with the single-frame 25 bit MSVQ from the 2.4 kb/s coder, the optimal interpolation patterns for the first two frames are vector quantized with four bits, and finally the remaining error of these two frames is jointly quantized with 14 bit MSVQ. The Fourier magnitudes for the last frame are quantized with the same 8 bit VQ as in 2.4 kb/s, and the remaining frame magnitudes are regenerated using interpolation. The six gain values are vector quantized with 10 bits, and the four bandpass voicing decisions for each voiced frame are quantized with a two-bit codebook. The aperiodic flag is quantized with one bit per superframe, with codebooks selected by the overall voicing pattern.

In 2006, the NATO STANAG 4591 was extended to also operate at 600 b/s [16.76]. As with the 1.2 kb/s version, this coder is based on the 2.4 kb/s coder with a longer analysis superframe. In this case, four consecutive frames are grouped into a 90 ms superframe. For each superframe, the overall voicing and bandpass voicing decisions are first jointly vector quantized with five bits. The remaining quantization algorithms are mode-dependent, based on this voicing pattern. The pitch trajectory is quantized using up to 8 bits, using scalar quantization with adaptive interpolation. The LSFs are quantized using a matrix extension of MSVQ, with each matrix representing the concatenation of two consecutive input vectors. Finally, the eight gain values in a superframe are jointly quantized using either full VQ or MSVQ, depending on the mode. The aperiodic flag and Fourier magnitudes are not used at this low rate.

Overall, the NATO STANAG 4591 MELPe coder family provides communication-quality speech at bit rates of 2.4 kb/s, 1.2 kb/s, and 600 b/s. MELPe performance at 2.4 kb/s is better than 4.8 kb/s FS1016, and there is a graceful degradation in performance with decreasing bit rate. Even at 600 b/s, MELPe performance is still better than 2.4 kb/s LPC-10e [16.76].

16.5.3 Satellite Communications

There are a number of commercial satellite communication systems for use in situations where cellular telephony may be impractical, such as for maritime applications [16.77]. These include the Inmarsat system, as well as other satellite communication systems such as ICO Global Communications, Iridium, Asia Cellular Satellite (ACeS), Optus, and American Mo-

bile Satellite Corporation Telesat Mobile Incorporated (AMSC-TMI). While the published literature on speech coding standards in such systems is limited, many use versions of MBE tailored for bit rates around 4 kb/s, for example the IMBE coder described in [16.78] or the more-recent proprietary AMBE coder. These systems are designed to provide communications quality in potentially high bit error rates. According to one assessment [16.79], the AMBE coder intended for use in the Inmarsat Aeronautical system provides speech quality approximately equivalent to first-generation digital cellular (specifically the US time-division multiple-access (TDMA) full-rate standard vector sum excited linear prediction (VSELP) algorithm [16.80]).

In addition to these satellite communication systems, a similar technology has been standardized for North American digital land mobile radio systems by the Association of Public-Safety Communications Officials (APCO) Project 25.

16.5.4 ITU 4 kb/s Standardization

We conclude this section with a discussion of an effort by the telecommunications standardization sector of the International Telecommunications Union (ITU-T) to standardize a toll-quality 4 kb/s speech coder for a wide range of communication applications. The ambitious goals of this project were announced in 1994 [16.81].

Unfortunately, achieving these goals has proven more difficult than initially anticipated. In parametric low-rate coders, 4 kb/s improvements have focussed on better representation of the current model parameters, i. e., faster frame update, more-accurate modeling of spectral amplitudes, and the use of complex Fourier coefficients (or equivalently waveform phase information), as in [16.82–85].

Other approaches to the 4 kb/s problem abandoned the parametric model entirely and worked with higher-rate waveform coders, based on CELP, to lower their bit rate. Key to these approaches is the idea of relaxing the waveform matching constraint using the generalized analysis-by-synthesis coding paradigm, also known as RCELP [16.86]. Significant benefits are achieved by allowing the CELP synthesis to drift in time relative to the input signal according to the best pitch prediction path. This approach was taken in one of the most promising ITU 4 kb/s candidates [16.87].

A third alternative is a *hybrid* coder, using both waveform and parametric coding at different times [16.23, 49, 88, 89]. Since parametric coders are good at producing periodic voiced speech and waveform coders are good at modeling transitional frames, a hybrid coder uses each where it is best suited, and switches between the two as the speech characteristics changes. A hybrid MELP/CELP coder was the basis for the other top-performing candidate [16.90].

After almost a decade of effort for the ITU 4 kb/s standardization, it is clear that, while multiple low-rate coding approaches can sometimes produce very high speech quality in formal evaluations, it is difficult to achieve toll-quality performance in all conditions. The top candidates were able to improve their performance in multiple rounds of testing, but none were able to achieve toll quality reliably in all testing laboratories and languages, and as a result no ITU 4 kb/s standard has yet been named.

16.6 Summary

Low-rate speech coding can now provide reliable communications-quality speech at bit rates well below 4 kb/s. While there are a number of different approaches to achieve these rates, such as MELP, STC, MBE, and WI, all rely on flexible parametric models combined with sophisticated quantization techniques to achieve this performance. Successful standardization of low-rate coders has enabled their widespread use for military and satellite communications. However, the goal of toll-quality low-rate coding continues to provide a research challenge.

References

16.1 M.R. Schroeder, B.S. Atal: Code excited linear prediction (CELP): High quality speech at very low bit rates, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (Tampa 1985) pp. 937–940

16.2 L.R. Rabiner, R.W. Schafer: *Digital Processing of Speech Signals* (Prentice Hall, Englewood Cliffs 1978)

16.3 G. Fant: *Acoustic Theory of Speech Production* (Mouton, The Hague 1960)

16.4 L.E. Kinsler: *Fundamentals of Acoustics*, 3rd edn. (Wiley, New York 1982)

16.5 B. Scharf: Critical bands. In: *Foundations of Modern Auditory Theory*, ed. by J.V. Tobias (Academic, New York 1970), Chap. 5

16.6 D.O. Kim, W.S. Rhode, S.R. Greenberg: Responses of cochlear nucleus neurons to speech signals: Neural encoding of pitch, intensity, and other parameters. In: *Auditory Frequency Selectivity*, ed. by B.C.J. Moore, R.D. Patterson (Plenum, New York 1986) pp. 281–288

16.7 H. Dudley: Remaking speech, J. Acoust. Soc. Am. **11**, 169–177 (1939)

16.8 J.N. Holmes: The influence of glottal waveform on the naturalness of speech from a parallel formant synthesizer, IEEE Trans. Audio Electroacoust. **21**, 298–305 (1973)

16.9 D.H. Klatt: Review of text-to-speech conversion for english, J. Acoust. Soc. Am. **82**, 737–793 (1987)

16.10 F. Itakura, S. Saito: Analysis synthesis telephony based on the maximum likelihood method, Rep. 6th Int. Congr. Acoustics (1968) pp. C17–C20

16.11 B.S. Atal, S.L. Hanauer: Speech analysis and synthesis by linear prediction of the speech wave, J. Acoust. Soc. Am. **50**(2), 637–655 (1971)

16.12 O. Fujimura: An approximation to voice aperiodicity, IEEE Trans. Audio Electroacoust. **16**, 68–72 (1968)

16.13 J. Makhoul, R. Viswanathan, R. Schwartz, A.W.F. Huggins: A mixed-source model for speech compression and synthesis, J. Acoust. Soc. Am. **64**(6), 1577–1581 (1978)

16.14 S.Y. Kwon, A.J. Goldberg: An enhanced LPC vocoder with no voiced/unvoiced switch, IEEE Trans. Acoust. Speech Signal Process. **32**, 851–858 (1984)

16.15 G.S. Kang, S.S. Everett: Improvement of the excitation source in the narrow-band linear prediction vocoder, IEEE Trans. Acoust. Speech Signal Process. **33**, 377–386 (1985)

16.16 M.R. Sambur, A.E. Rosenberg, L.R. Rabiner, C.A. McGonegal: On reducing the buzz in LPC synthesis, J. Acoust. Soc. Am. **63**, 918–924 (1978)

16.17 D.Y. Wong: On understanding the quality problems of LPC speech, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1980) pp. 725–728

16.18 B.S. Atal, N. David: On synthesizing natural-sounding speech by linear prediction, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1979) pp. 44–47

16.19 A. McCree, T.P. Barnwell III: A mixed excitation LPC vocoder model for low bit rate speech coding, IEEE Trans. Speech Audio Process. **3**(4), 242–250 (1995)

16.20 A. McCree, T.P. Barnwell III: Improving the performance of a mixed excitation LPC vocoder in acoustic noise, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1992) pp. II137–II140

16.21 W. Hess: *Pitch Determination of Speech Signals* (Springer, Berlin, Heidelberg 1983)

16.22 A. McCree, T.P. Barnwell III: A new mixed excitation LPC vocoder, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1991) pp. 593–596

16.23 D.L. Thomson, D.P. Prezas: Selective modeling of the LPC residual during unvoiced frames: White noise or pulse excitation, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1986) pp. 3087–3090

16.24 J.H. Chen, A. Gersho: Real-time vector APC speech coding at 4800 bps with adaptive postfiltering, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1987) pp. 2185–2188

16.25 W.B. Kleijn, D.J. Krasinski, R.H. Ketchum: Fast methods for the CELP speech coding algorithm, IEEE Trans. Acoust. Speech Signal Process. **38**(8), 1330–1342 (1990)

16.26 J.N. Holmes: Formant excitation before and after glottal closure, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1976) pp. 39–42

16.27 A.E. Rosenberg: Effect of glottal pulse shape on the quality of natural vowels, J. Acoust. Soc. Am. **49**, 583–590 (1971)

16.28 A. McCree, J.C. DeMartin: A 1.7 kb/s MELP coder with improved analysis and quantization, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1998) pp. 593–596

16.29 T. Unno, T.P. Barnwell III, K. Truong: An improved mixed excitation linear prediction (MELP) coder, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1999) pp. 245–248

16.30 W. Lin, S.N. Koh, X. Lin: Mixed excitation linear prediction coding of wideband speech at 8 kbps, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (2000) pp. II1137–II1140

16.31 N.R. Chong-White, R.V. Cox: An intelligibility enhancement for the mixed excitation linear prediction speech coder, IEEE Signal Process. Lett. **10**(9), 263–266 (2003)

16.32 A.E. Ertan, T.P. Barnwell III: Improving the 2.4 kb/s military standard MELP (MS-MELP) coder using pitch-synchronous analysis and synthesis techniques, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (2005) pp. 761–764

16.33 R.J. McAulay, T.F. Quatieri: Sinusoidal coding. In: *Speech Coding and Synthesis*, ed. by W.B. Kleijn, K.K. Paliwal (Elsevier, Amsterdam 1995), Chap. 4

16.34 T.F. Quatieri: *Discrete Time Speech Signal Processing: Principles and Practice* (Prentice Hall, Englewood Cliffs 2002), Chap. 9

16.35 P. Hedelin: A tone-oriented voice-excited vocoder, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1981) pp. 205–208

16.36 L.B. Almeida, F.M. Silva: Variable-frequency synthesis: An improved harmonic coding scheme, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1984), Sects. 27.5.1–27.5.4.

16.37 R.J. McAulay, T.F. Quatieri: Speech analysis/synthesis based on a sinusoidal representation, IEEE Trans. Acoust. Speech Signal Process. **ASSP-34**(4), 744–754 (1986)

16.38 D.W. Griffin, J.S. Lim: Multiband excitation vocoder, IEEE Trans. Acoust. Speech Signal Process. **36**(8), 1223–1235 (1988)

16.39 E.B. George, M.J.T. Smith: Speech analysis/synthesis and modification using an analysis-by-synthesis/overlap-add sinusoidal model, IEEE Trans. Speech Audio Process. **5**(5), 389–406 (1997)

16.40 C. Li, V. Cuperman: Analysis-by-synthesis multimode harmonic speech coding at 4 kb/s, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 3 (2000) pp. 1367–1370

16.41 C.O. Etemoglu, V. Cuperman, A. Gersho: Speech coding with an analysis-by-synthesis sinusoidal model, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 3 (2000) pp. 1371–1374

16.42 M.S. Brandstein: *A 1.5 kbps multi-band excitation speech coder, M.S. Thesis* (Massachusetts Institute of Technology, Cambridge 1990)

16.43 R. McAulay, T. Parks, T. Quatieri, M. Sabin: Sine-wave amplitude coding at low data rates. In: *Advances in Speech Coding*, ed. by B.S. Atal, V. Cuperman, A. Gersho (Kluwer Academic, Boston 1991) pp. 203–214

16.44 S. Yeldener, A.M. Kondoz, B.G. Evans: High quality multiband LPC coding of speech at 2.4 kbit/s, Electron. Lett. **27**(14), 1287–1289 (1991)

16.45 M. Nishiguchi, J. Matsumoto, R. Wakatsuki, S. Ono: Vector quantized MBE with simplified V/UV division at 3.0 kbit/s, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 2 (1993) pp. 151–154

16.46 A. Das, A.V. Rao, A. Gersho: Variable-dimension vector quantization, IEEE Signal Process. Lett. **3**(7), 200–202 (1996)

16.47 P. Lupini, V. Cuperman: Nonsquare transform vector quantization, IEEE Signal Process. Lett. **3**(1), 1–3 (1996)

16.48 W.B. Kleijn, J. Haagen: Waveform interpolation for coding and synthesis. In: *Speech Coding and Synthesis*, ed. by W.B. Kleijn, K.K. Paliwal (Elsevier, Amsterdam 1995), Chap. 5

16.49 W.B. Kleijn: Encoding speech using prototype waveforms, IEEE Trans. Speech Audio Process. **1**(4), 386–399 (1993)

16.50 W.B. Kleijn, J. Haagen: Transformation and decomposition of the speech signal for coding, IEEE Signal Process. Lett. **1**, 136–138 (1994)

16.51 T. Eriksson, W.B. Kleijn: On waveform-interpolation coding with asymptotically perfect reconstruction, Proc. IEEE Workshop on Speech Coding (1999) pp. 93–95

16.52 N.R. Chong, I.S. Burnett, J.F. Chicharo: A new waveform interpolation coding scheme based on pitch synchronous wavelet transform decomposition, IEEE Trans. Speech Audio Process. **8**(3), 345–348 (2000)

16.53 A. Gersho, R.M. Gray: *Vector Quantization and Signal Compression* (Kluwer, Dordrecht 1992)

16.54 H. Dudley: Phonetic pattern recognition vocoder for narrow-band speech transmission, J. Acoust. Soc. Am. **30**, 733–739 (1958)

16.55 C.E. Shannon: A mathematical theory of communication, Bell Syst. Tech. J. **27**, 379–423,623–656 (1948)

16.56 J. Picone, G. Doddington: A phonetic vocoder, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1989) pp. 580–583

16.57 J. Makhoul: Linear prediction: A tutorial review, IEEE Proc. **63**, 561–579 (1975)

16.58 F. Itakura: Line spectrum representation of linear predictive coefficients of speech signals, J. Acoust. Soc. Am. **57**, S35(A) (1975)

16.59 K.K. Paliwal, B.S. Atal: Efficient vector quantization of LPC parameters at 24 bits/frame, IEEE Trans. Speech Audio Process. **1**(1), 3–14 (1993)

16.60 J.S. Collura, A. McCree, T.E. Tremain: Perceptually based distortion measures for spectrum quantization, Proc. IEEE Workshop on Speech Coding for Telecommunications (1995) pp. 49–50

16.61 W.P. LeBlanc, B. Bhattacharya, S.A. Mahmoud, V. Cuperman: Efficient search and design procedures for robust multi-stage VQ of LPC parameters for 4 kb/s speech coding, IEEE Trans. Speech Audio Process. **1**(4), 373–385 (1993)

16.62 W. Gardner, B. Rao: Theoretical analysis of the high-rate vector quantization of LPC parameters, IEEE Trans. Speech Audio Process. **3**, 367–381 (1995)

16.63 A. McCree, J.C. DeMartin: A 1.6 kb/s MELP coder for wireless communications, Proc. IEEE Workshop on Speech Coding for Telecommunications (1997) pp. 23–24

16.64 T.E. Tremain: The government standard linear predictive coding algorithm: LPC-10, Speech Technol. **1**, 40–49 (1982)

16.65 T.E. Tremain, M.A. Kohler, T.G. Champion: Philosophy and goals of the DoD 2400 bps vocoder selection process, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 2 (1996) pp. 1137–1140

16.66 A. McCree, K. Truong, E.B. George, T.P. Barnwell III, V.R. Viswanathan: A 2.4 kbit/s MELP coder candidate for the new U.S. Federal Standard, Proc. IEEE Int. Conf. Acoust. Speech Signal Processing, Vol. 1 (1996) pp. 200–203

16.67 L.M. Supplee, R.P. Cohn, J.S. Collura, A. McCree: MELP: the new Federal Standard at 2400 bps, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 2 (1997) pp. 1591–1594

16.68 M.A. Kohler: A comparison of the new 2400 bps MELP federal standard with other standard coders,

Proc. IEEE Int. Conf. Acoust. Speech Signal Processing (1997) pp. 1587–1590
16.69 J.P. Campbell Jr., T.E. Tremain, V.C. Welch: The DoD 4.8 kbps Standard (Proposed Federal Standard (1016)). In: *Advances in Speech Coding*, ed. by B.S. Atal, V. Cuperman, A. Gersho (Kluwer Academic, Boston 1991) pp. 121–133
16.70 S. Villette, K.T. Al Naimi, C. Sturt, A.M. Kondoz, H. Palaz: A 2.4/1.2 kbps SB-LPC based speech coder: the Turkish NATO STANAG candidate, Proc. IEEE Workshop on Speech Coding (2002) pp. 87–89
16.71 G. Guilmin, P. Gournay, F. Chartier: Description of the French NATO candidate, Proc. IEEE Workshop on Speech Coding (2002) pp. 84–86
16.72 T. Wang, K. Koishida, V. Cuperman, A. Gersho, J.S. Collura: A 1200/2400 bps coding suite based on MELP, Proc. IEEE Workshop on Speech Coding (2002) pp. 90–92
16.73 J.S. Collura, D.F. Brandt, D.J. Rahikka: The 1.2 kbps/2.4 kbps MELP speech coding suite with integrated noise pre-processing, IEEE Mil. Commun. Conf. Proc., Vol. 2 (1999) pp. 1449–1453
16.74 R. Martin, R.V. Cox: New speech enhancement techniques for low bit rate speech coding, Proc. IEEE Workshop on Speech Coding (1999) pp. 165–167
16.75 T. Wang, K. Koishida, V. Cuperman, A. Gersho, J.S. Collura: A 1200 bps speech coder based on MELP, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (2000) pp. 1375–1378
16.76 G. Guilmin, F. Capman, B. Ravera, F. Chartier: New NATO STANAG narrow band voice coder at 600 bits/s, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (2006) pp. 689–693
16.77 J.V. Evans: Satellite systems for personal communications, Proc. IEEE, Vol. 86 (1998) pp. 1325–1341
16.78 J.C. Hardwick, J.S. Lim: The application of the IMBE speech coder to mobile communications, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 1 (1991) pp. 249–252
16.79 S.F.C. Neto, F.L. Corcoran, J. Phipps, S. Dimolitsas: Performance assessment of 4.8 kbit/s AMBE coding under aeronautical environmental conditions, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 1 (1996) pp. 499–502
16.80 I.A. Gerson, M.A. Jasiuk: VSELP). In: *Advances in Speech Coding*, ed. by B.S. Atal, V. Cuperman, A. Gersho (Kluwer, Norwell 1991) pp. 69–79
16.81 S. Dimolitsas, C. Ravishankar, G. Schroder: Current objectives in 4-kb/s wireline-quality speech coding standardization, IEEE Signal Process. Lett. **1**(11), 157–159 (1994)
16.82 E.L.T. Choy: *Waveform interpolation speech coder at 4 kb/s*, M.S. Thesis (McGill University, Montreal 1998)
16.83 O. Gottesman, A. Gersho: Enhanced waveform interpolative coding at low bit-rate, IEEE Trans. Speech Audio Process. **9**(8), 786–798 (2001)
16.84 J. Stachurski, A. McCree, V. Viswanathan: High quality MELP coding at bit-rates around 4 kb/s, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1999) pp. 485–488
16.85 S. Yeldener: A 4 kb/s toll quality harmonic excitation linear predictive speech coder, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 1 (1999) pp. 481–484
16.86 W.B. Kleijn, R.P. Ramachandran, P. Kroon: Generalized analysis-by-synthesis coding and its application to pitch rediction, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 1 (1992) pp. 337–340
16.87 J. Thyssen, Y. Gao, A. Benyassine, E. Shlomot, C. Murgia, H. Su, K. Mano, Y. Hiwasaki, H. Ehara, K. Yasunaga, C. Lamblin, B. Kovesi, J. Stegmann, H. Kang: A candidate for the ITU-T 4 kbit/s speech coding standard, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 2 (2001) pp. 681–684
16.88 I.M. Trancoso, L. Almeida, J.M. Tribolet: A study on the relationships between stochastic and harmonic coding, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1986) pp. 1709–1712
16.89 E. Shlomot, V. Cuperman, A. Gersho: Combined harmonic and waveform coding of speech at low bit rates, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1998) pp. 585–588
16.90 A. McCree, J. Stachurski, T. Unno, E. Ertan, E. Paksoy, V. Viswanathan, A. Heikkinen, A. Ramo, S. Himanen, P. Blocher, O. Dressler: A 4 kb/s hybrid MELP/CELP speech coding candidate for ITU standardization, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 1 (2002) pp. 629–632

18. Perceptual Audio Coding of Speech Signals

J. Herre, M. Lutzky

Traditionally algorithms for speech coding exploit the features of speech signals by employing algorithmic models of the human vocal tract. More recently, the use of generic audio coders for coding of speech signals has gained increasing importance. Based on the properties of human hearing, such perceptual audio coders offer attractive properties including full-bandwidth audio output, increased naturalness, and good handling of any type of non-speech material. The chapter discusses the principles of perceptual audio coding, some relevant standards, and a number of perceptual audio coders that find application in speech and audio transmission and storage.

18.1 History of Audio Coding

Historically, source coding of speech was motivated by the human desire for communication with other individuals over distance – commercially speaking: how to get as many speech connections across a given telephone network as possible while maintaining high intelligibility and good subjective voice quality. As a result, the field of speech coding evolved very much with a focus on real-time two-way communication and a sound fidelity which is adapted to the needs of this application area. Accordingly, the technical approach adopted by typical speech coders reflects and exploits the specific process of speech production in the human vocal tract (see Part A of this Handbook) to a large extent.

Much later (around 1980), people started thinking about the efficient transmission and storage of *music* and *generic audio* signals. The fact that music constitutes a much broader class of signals than speech, and does not adhere to a common model of generation, led to very different technical solutions to the audio coding problem as compared to speech coding. Typically, audio coding is thus strongly motivated by the quest to exploit perceptual properties of the final receiver, i. e., the human auditory system, in order to provide maximum (ideally: indistinguishable) subjective quality across a broad range of signals. These ideas are reflected by the concept of *perceptual audio coding* for *general audio signals*. Naturally, perceptual audio coders are also able to code speech signals, and thus serve for a range of speech-related applications. Moreover, as more bandwidth becomes gradually available, using perceptual audio coders for speech coding offers several clear benefits over traditional speech coders,

such as the ability to convey the whole audible signal bandwidth, the associated increase in naturalness and intelligibility, the capability to represent also speech with arbitrary background sounds (including music), handling of stereo/multichannel signals etc.

This chapter provides some background about the field of perceptual audio coding and touches upon several relevant coding standards. Subsequently, a number of perceptual audio coders are discussed which find application in speech and audio transmission and storage.

18.2 Fundamentals of Perceptual Audio Coding

18.2.1 General Background

The basic task of an audio coder can be summarized briefly: the (uncompressed) source audio data should be represented as compactly as possible while maintaining its original sound quality. Ideally, the decompressed audio signal should sound identical to the original one under all circumstances (*perceptual transparency*). To this end, the paradigm of *perceptual audio coding* turns out to be an extremely powerful approach which aims at optimizing the subjectively perceived audio quality rather than other objective distortion metrics, such as mean-squared-error (MSE) and signal-to-noise ratio (SNR). This is achieved by employing knowledge about human perception of sound as represented by the field of psychoacoustics. In particular, the use of this knowledge allows to exploit the limits of perceptibility for coding distortion and achieve transparent coding (if the incurred coding distortion is below the threshold of perceptibility) or at least minimize the subjective disturbance caused by the coding distortion at low bit rates.

As the range of input signals for an audio coder is potentially unlimited (just think of the range of possible acoustic and electronic musical instruments, environmental sounds etc.), no universal source model is available as a basis for the coding process. This situation is very different from, e.g., speech coding, where a well-grounded model of the human vocal tract can be used as a source (production) model of the input signal, and thus facilitates the exploitation of its inherent redundancy. As a consequence, perceptual audio coders have to be extremely flexible so as to cover the wide range of potential input signals. Moreover, they have to rely significantly on the properties of the final signal receiver, i. e., the human auditory system, exploiting signal irrelevance. In particular, the so-called masking effect describes the phenomenon that louder signal components conceal the presence of other, weaker components which are close in frequency or time (see, e.g., [18.1, 2]).

18.2.2 Coder Structure

Although there may be many differences in technical details among current audio compression schemes, most common coders lean on the basic paradigm of *filterbank-based audio coding*. This may be explained by the fact that a filterbank-based coder framework combines both the ability of reducing redundancy and exploiting the potential behind the removal of irrelevancy. On the one hand, coding of a spectral representation is an efficient means to take advantage of linear correlation between subsequent samples of an input signal. Redundancy can be further reduced by entropy coding of the spectral coefficients. On the other hand, a spectral representation of the input signal opens the door to a good first-order modeling of the limits of human auditory perception. Once the threshold of perceptibility (audibility) is estimated, the precision of transmission can be easily adapted to match the threshold by adjusting the quantization distortion of the corresponding spectral coefficients.

The upper part of Fig. 18.1 depicts the well-known basic block diagram of a generic monophonic perceptual audio coder which comprises the following components:

- *Analysis Filterbank:* The input signal is mapped to a subsampled spectral representation using various types of analysis filterbanks. For reasons of coding efficiency, modern coding schemes typically employ filterbanks with critical sampling (i. e., same number of input samples and spectral coefficients) and overlapping analysis windows between subsequent analysis frames. Examples include the modified discrete cosine transform (MDCT) [18.3], polyphase filterbanks [18.4], or hybrid structures [18.5]. An account of common filterbanks for perceptual audio coding can be found in [18.6].
- *Perceptual model:* the signal's time- and frequency-dependent threshold of perceptibility (masking threshold) is estimated by a perceptual model. This value describes the maximum quantization error that

can be introduced into the audio signal while still maintaining perceptually unimpaired signal quality. Typically, perceptual models for audio coding consider psychoacoustic effects like masking in frequency domain (inter- and intraband masking), temporal masking effects (postmasking) and the asymmetry of masking between tonal and noise-like stimuli [18.7]. In order to match closely the properties of the peripheral human auditory system, masking calculations are usually performed on a nonuniform frequency scale, such as the BARK or ERB scale [18.1, 2].

- *Quantization and coding:* the spectral values are quantized and coded with a precision corresponding to the masking threshold estimate. Both uniform and nonuniform scalar quantization and vector quantization have been used in perceptual audio coding. Additional reduction of redundancy can be achieved by employing entropy coding techniques, such as Huffman coding. Usually, the quantization/coding kernel of an encoder operates under two constraints. While the estimated masking threshold defines a target for the minimum precision of the coded signal representation, coders frequently operate under the limitations of a fixed bit rate selected by the user. As it may not be possible to satisfy both constraints at the same time, a good encoding algorithm has to produce an encoding solution representing an acceptable compromise with respect to perceived audio quality in such cases. Besides the perceptual modeling aspect, these strategies for *bit allocation/noise allocation* are essential parts of an optimized audio coder [18.8].
- *Bitstream encoding:* finally, all relevant information (i. e. the coded spectral values and additional side information) is packed into a bitstream and transmitted to the decoder.

Correspondingly, the processing steps appear in reversed order in the decoder (see the lower part of Fig. 18.1). The bitstream is decoded and parsed into coded spectral data and side information. The inverse quantization of the quantized spectral coefficients is then carried out. Finally, the spectral values are mapped back to a time-domain representation using a synthesis filterbank.

While all common filterbank-based perceptual audio coders are based on this generic structure, practical coders are extended by a number of optional building blocks (*coding tools*) which provide further improvements in coding performance for specific coding constellations. These include provisions for joint coding of stereo signals, enhanced exploitation of redundancy (e.g., by means of predictors) or irrelevance (by enhancing the coder's noise-shaping abilities).

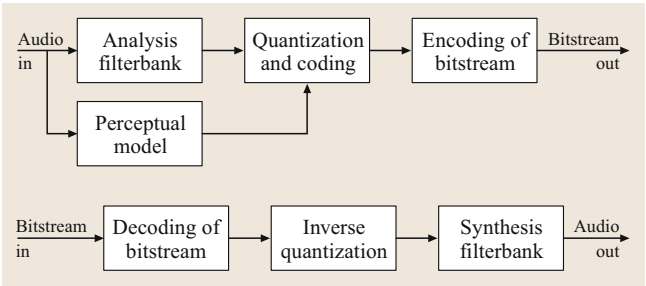


Fig. 18.1 Generic structure of a monophonic perceptual audio coder/decoder

18.2.3 Perceptual Audio Coding Versus Speech Coding

When comparing the concepts behind speech coding and perceptual audio coding, several aspects are to be considered. While efficient speech coding is very much focused on exploiting a model of the signal source (i. e., the human vocal tract), perceptual audio coders can not rely on such an assumption but have to resort to generic means to take advantage of source characteristics (i. e., spectral decomposition and entropy coding). On the other hand, the perceptual audio coding approach allows to benefit substantially from our knowledge about the signal receiver (i. e., the human auditory system) by shaping the introduced coding noise such that its audibility is minimized by auditory masking in frequency and time. Compared to this, the ability to exploit masking effects in a traditional speech coding framework [e.g., by perceptual weighting of the coding error in code-excited linear prediction (CELP) schemes] appears much less pronounced.

Another traditional difference between speech and audio coding lies in their supported audio bandwidth. While music signals contain perceptually significant components up to at least 16 kHz and thus require a correspondingly high bandwidth for an unimpaired listening impression from audio coders, the foremost goal of speech coding for voice communication has been *intelligibility*. As a consequence, communication standards often are based on *narrow-band* audio channels (300 Hz–3.4 kHz). Unfortunately, some essential information of the human voice is located beyond this frequency range. As an example, the consonants *f* and *s* have the same spectral composition in a narrow-band

representation, which leads to an equal sounding of the words *failing* and *sailing* when transmitted by a narrow-band speech coder. A differentiation is only possible for a listener from the semantic message context. In order to make long conversations less exhausting, *wide-band* speech coders (50 Hz–7 kHz) gain importance. Even this extended frequency range, however, does not reproduce the entire audible spectrum, and therefore discards relevant auditory information. Due to their focus on music, perceptual audio coders traditionally provide a full bandwidth transmission at high enough bit rates, and thus offer optimal quality also in very difficult communication situations (e.g., simultaneous interpretation where at least 12 kHz bandwidth is demanded).

18.3 Some Successful Standardized Audio Coders

This section provides a sketch of several successful standardized perceptual audio coders that have evolved over the recent two decades rather than a comprehensive technical account. In this context, the moving pictures expert group (MPEG), formally International Organization for Standardization/International Electrotechnical Commission (ISO/IEC) JTC1/SC29/WG11, standardization group plays a major role. On the one hand, the first widely available commercial audio codecs were developed by this group and found broad application. On the other hand, subsequent generations of MPEG audio coding schemes have continued to define the state of the art until today. Thus, the evolution of MPEG audio coding will be taken as an illustration of the general development in this field, acknowledging the presence of other proprietary coders which have been successful in the marketplace for considerable time.

18.3.1 MPEG-1

In 1992, MPEG finalized its first international standard on coding of moving pictures and associated audio, called MPEG-1 [18.9]. The audio part of this standard (ISO/IEC 11172-3) specifies a generic standard for the coding of general audio signals (as opposed to speech signals as used in telecommunications / speech applications) and became the driving technology for many important applications relying on audio compression. The standard defines three operating modes (*layers*) in order to meet the requirements of a broad range of applications of audio coding concerning target bit rate and complexity:

The layer 2 coding scheme defines a codec with medium compression performance and computational complexity which had its first applications in the broadcasting environment, including use for the European digital audio broadcasting (DAB) system. The codec achieves broadcast quality at a bit rate of 128 kbit/s per channel.

Relative to this, the layer 1 coding scheme represents a simplified version of the layer 2 coder with both lower computational complexity and compression. The intended major application for this scheme was digital recording of audio for consumers [the digital compact cassette (DCC)]. Layer 1 reaches broadcast quality at a bit rate of 192 kbit/s per channel.

In contrast to this, the layer 3 coding scheme was designed to provide the best possible compression performance by more-refined and computationally demanding coding techniques. Audio transmission via integrated services digital network (ISDN) channels and satellite radio were the first major applications of this codec. Layer 3 has become very popular in the context of Internet and personal audio under the name MP3 . The codec offers broadcast quality at a bit rate of 96 kbit/s per channel.

18.3.2 MPEG-2

While the coding schemes provided by MPEG-1 Audio were developed to meet the needs of most high-quality applications, more capabilities were added to the existing three Layer coder family by the MPEG-2 audio standard [18.10], which was finalized in 1994. Most notably, MPEG-2 provides the specification for coding at low sampling rates (LSR), expanding the set of standard sampling rates supported by MPEG-1 audio (i. e., 32 kHz, 44.1 kHz, and 48 kHz) by their half-rate counterparts (i. e., 16 kHz, 22.05 kHz, 24 kHz). Though this leads to clear limitations in the obtainable audio bandwidth, it results in an increase in coding efficiency for low bit rates and a reduction of computational demands. Typically, MPEG-2 low-sampling-rate coding is used in multimedia applications, e.g., in desktop personal computers (PCs= where the best possible quality may not be of utmost concern.

As a second less well-known extension, the MPEG-2 specification defines backward compatible multichan-

nel counterparts for the MPEG-1 audio coder family. MPEG-2 multichannel audio bitstreams can be decoded by MPEG-1 audio decoders.

18.3.3 MPEG-2 Advanced Audio Coding

After the completion of the MPEG-2 multichannel audio specification in 1994, the MPEG audio standardization group initiated another effort to define an enhanced multichannel coding standard which should not be constrained by the requirement of MPEG-1 backward compatibility. The standardization process was successfully completed in 1997 and its result is known as MPEG-2 advanced audio coding (MPEG-2 AAC), which became an addendum to the MPEG-2 standard (ISO/IEC 13818-7) [18.11]. While the new scheme had been initially developed as a multichannel audio coder, further tests confirmed its coding performance also for stereo and monophonic signals with broadcast quality according to ITU radiocommunication sector (ITU-R) requirements at a bit rate of only 64 kbit/s per channel. Furthermore, the coder can be used within a wide range of sampling rates (8–96 kHz) and data rates (from below 16 kbit/s to 128 kbit/s per audio channel).

18.3.4 MPEG-4 Advanced Audio Coding

While the MPEG-4 audio specification provides a rich set of different specialized (e.g., speech) and generic coders for all conceivable types of applications [18.12, 13], the MPEG-2 AAC technology was adopted as the core part of MPEG-4 *general audio coding* (i. e., the filterbank-based MPEG-4 perceptual audio coder) and augmented with a number of extensions and functionalities. In the context of coding speech signals, several of these extensions are of particular interest:

- *MPEG-4 scalable audio coding:* one of the core functionalities provided by MPEG-4 Audio is called *scalability*, i. e., the ability of decoding only part of the full bitstream into a meaningful result, albeit with a lower perceptual quality. MPEG-4 scalable audio bitstreams consist of several embedded *layers* of information each of which originates from a partial coder. Interestingly, MPEG-4 Audio defines also coder configurations which include both speech coders and perceptual audio coders, and thus combine both concepts into a single system. Such systems will be described in some more detail in Sect. 18.5.1.
- *MPEG-4 low-delay AAC:* another requirement for MPEG-4 Audio coding was the ability to provide full-quality general audio coding with an encoding/decoding delay that is comparable to usual speech codecs and, thus is low enough to enable high-quality two-way communication (e.g., teleconferencing). This goal is achieved by a derivative of AAC, called MPEG-4 low-delay AAC. The concepts behind this codec will be described in Sect. 18.4.2.
- *Bandwidth extension:* the concept of bandwidth extension [18.14, 15] is one of the most attractive recent additions to the MPEG-4 audio coding since it provides a notable increase in coding efficiency at low bit rates/intermediate audio quality and, at the same time, removes the traditional problem of band-limited audio output at low bit rates. This technique allows to reconstruct a good perceptual approximation of the signal's original high-frequency content (e.g., above 5–6 kHz) by transmitting its low-frequency part together with a small amount of side information representing the high-frequency spectral envelope. The combination of an AAC coder with bandwidth extension, also known as *high-efficiency AAC (HE-AAC)*, achieves full bandwidth output with attractive sound quality at 24 kbit/s per audio channel and below.
- *Parametric Stereo:* Another step towards even higher coding efficiency for MPEG-4 AAC-based coders is the combination with the *parametric stereo (PS)* coding tool that has been developed as part of the MPEG-4 high-quality parametric audio coding technology [18.16, 17]. The tool allows to efficiently represent a stereo signal by transmitting a monophonic downmix and a compact parametric side information which carries a perceptual characterization of the stereophonic sound image. Based on this description, the parametric stereo decoder expands the transmitted mono downmix signal into a stereophonic output. Combined with a HE-AAC coder, good sound quality can be achieved at 32 kbit/s per stereo channel and lower.

18.3.5 Progress in Coding Performance

As the original focus of audio coding has been on achieving highest possible compression while preserving subjective audio quality, it is quite natural to ask what improvements have been made in compression in the recent decades. Figure 18.2 attempts to provide an estimate of the bit rate (in kbit/s) needed to code a typical (not: critical) stereo signal in *good* quality, i. e.,

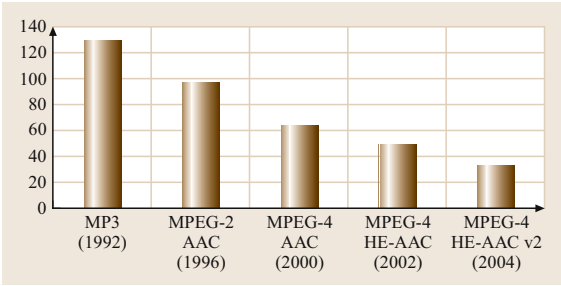


Fig. 18.2 Bit rates for *good-quality* stereo coding (in kbit/s)

in a sonic quality that is acceptable to most users for everyday consumption. Though this is admittedly a very loose definition, the *learning curve* shown in the figure is still instructive, showing a clear reduction in bit rate from 128 kbit/s (MP3) to 32 kbit/s (HE-AAC v2 = AAC plus bandwidth extension plus parametric stereo), i. e., by 75%. Evidently, there has been significant progress in the area of nontransparent coding, and it is interesting to see how much further the frontiers can be pushed in the future.

With respect to *transparent* coding of audio signals (including critical test signals), however, no comparable increase in compression efficiency has been encountered and coding at 128 kbit/s per stereo signal using MPEG-2 AAC can still be considered state of the art at the time of writing of this text.

18.4 Perceptual Audio Coding for Real-Time Communication

The preceding sections discussed technology that can be employed for perceptual coding of speech signals in general storage or transmission systems (e.g., radio or Internet radio systems, digital media, voice storage in announcement systems) for which overall delay does not play a major role. For application in two-way communication (e.g., telephony, teleconferencing, telepresence), however, a low overall delay is of vital importance to the quality of the service. For such applications the encoder/decoder (*codec*) must typically not introduce more than 20–30 ms of algorithmic delay, i. e., latency that is inherent in the algorithm disregarding speed of calculation and bitstream transmission. This limitation in codec delay is necessary in order to achieve an end-to-end transmission delay of typically no more than 150 ms in order to be acceptable for interactive speech communication. The actually observed end-to-end delay, however, also includes a number of additional delay sources, such as the delay introduced by the transmission network (e.g., by buffering within network switches or transcoding operations by network gateways) or audio postprocessing. This section analyzes the typical sources of codec delay in perceptual audio coding and discusses a number of low delay perceptual audio coding technologies that are suitable for real-time two-way speech communication.

18.4.1 Delay Sources in Perceptual Audio Coding

Generally, the goal of good compression efficiency can only be achieved at the expense of a certain encoding/decoding delay. The overall delay incurred by a typical perceptual audio coder results from several contributions related to the following codec parameters:

- Filterbank and framing delay
- Look-ahead time for block switching
- Algorithmic transmission delay

This section will discuss the background of these different delay contributions. As an example, the delay of an MPEG-2 AAC codec will be calculated in order to give some impression about the order of magnitude of delay for a modern high-performance perceptual codec. All calculations are based on the so-called *algorithmic delay* which describes the theoretical minimum delay allowed by an algorithm assuming negligible delay contributions due to speed of calculation, bitstream transmission or other implementation or application specific circumstances.

Filterbank and Framing Delay

In order to exploit the spectral masking properties of the human auditory system, perceptual audio coding schemes employ an analysis/synthesis filter bank pair. While numerous types of filter banks have been used for audio coding, the modified discrete cosine transform (MDCT) [18.3] has been adopted extensively for modern audio codecs, like MPEG-2 AAC and MPEG-4, and has shown its merits for compression at very low bit rates. For an MDCT with an overlap of 50% between subsequent symmetric windows, the filterbank introduces a total delay equal to its window size, i. e., twice the

frame size.

$$N_{\mathrm{MDCT}} = 2 \cdot \text{frame-size} \,. \tag{18.1}$$

Look-Ahead Delay for Block Switching Decision

Most modern audio codecs that are designed to operate at very low bit rates use an MDCT filterbank with a high spectral resolution. In particular, high efficiency can be reached with long frames (about 20 ms and more) for stationary signals. In the case of transient signals, like percussive sounds with sharp signal onsets (*attacks*), this would lead to the well-known pre-echo phenomenon [18.8]. This artifact can be avoided by using dynamic block switching [18.18], i. e., by dynamically switching between different filterbank window sizes, and thus reducing the blockwise noise spread in time. Due to restrictions in the permissible sequence of window types, *instantaneous* switching between long and short windows is not possible but an intermediate transition window type (*start block*) has to be inserted between long and short windows. Therefore, the detection of the optimum window type requires a look-ahead, and thus a further delay in the encoder. In general, an encoder using block switching incurs an additional delay of

$$N_{\mathrm{ahead}} = \text{frame-size} \cdot \frac{\text{numShortWindows} + 1}{2 \cdot \text{numShortWindows}} \,, \tag{18.2}$$

where numShortWindows is the number of short windows that fit into a frame (e.g., eight for AAC).

Algorithmic Transmission Delay

Since not all segments of an audio signal are equally demanding to code, the number of bits needed to code a specific frame varies. To obtain a constant bit rate, the bit reservoir mechanism has proven to be useful [18.8]. Since the use of the bit reservoir is equivalent to a local variation in bit rate, the size of the input buffer of the decoder must be adapted to the maximum local bit rate (i. e., the maximum number of bits which can be allocated for a single frame per channel). For the practically important case of a transmission on a channel with a rate equal to the encoding bit rate, use of the bit reservoir introduces an additional delay since the decoder has to wait at least until its input buffer is read before audio output can be started. Thus, increasing the size of the bit reservoir will also increase the overall codec delay. In fact, the overall delay of the audio coder may be dominated entirely by the size of the bit reservoir.

The delay expressed in terms of samples caused by the bit reservoir is

$$N_{\mathrm{bitres}} = \frac{\text{bitres-size}}{\text{bitrate}} F_{\mathrm{s}} \,, \tag{18.3}$$

where bitres-size is the bit reservoir size expressed in bits and F_{s} is the sampling rate in Hz. In addition, the time to transmit an average size frame has to be taken into account for delay considerations:

$$N_{\mathrm{frame}} = \frac{\text{average-framelength}}{\text{bitrate}} F_{\mathrm{s}} \,. \tag{18.4}$$

Note that in systems using packet-based transmission (one complete encoder frame per packet) with a channel bit rate much higher than the encoding bit rate, the transmission (and the use of the bit reservoir) does not cause the additional delay discussed above.

Overall Delay

From the discussion above, the overall delay of a typical coder can be calculated as follows:

$$t_{\mathrm{delay}} = \frac{N_{\mathrm{MDCT}} + N_{\mathrm{ahead}} + N_{\mathrm{bitres}} + N_{\mathrm{frame}}}{F_{\mathrm{s}}} \,, \tag{18.5}$$

with F_{s}: the coder sampling rate (in Hz), N_{MDCT}: the filterbank and framing delay, N_{ahead}: look-ahead delay for block switching, N_{bitres}: delay due to bit reservoir use, and N_{frame}: frame transmission delay.

Note that the overall delay scales inversely with the sampling frequency.

For an AAC coder running at 24 kbit/s per channel and a sampling rate of 24 kHz, the resulting overall codec delay for a non packet based transmission is about 109.3 ms without the use of the bit reservoir. Assuming the nominal size of the input buffer as indicated in the MPEG-2 AAC standard (6144 bit/channel), a maximum additional delay of 256 ms is incurred, leading to a total delay of 365.3 ms. From this example it becomes obvious that the delay of common perceptual audio coders substantially exceeds the requirements for real-time two-way communication applications.

18.4.2 MPEG-4 Low-Delay AAC

The MPEG-4 error resilient low-delay advanced audio (ER AAC-LD) coder [18.19] was derived from the architecture of MPEG-2/4 AAC to achieve the desired low delay required for two-way real-time communications applications with a minimum number of changes. More specifically, the low-delay codec is derived from the MPEG-4 general audio coder, which comprises the regular MPEG-2 AAC algorithm plus additions provid-

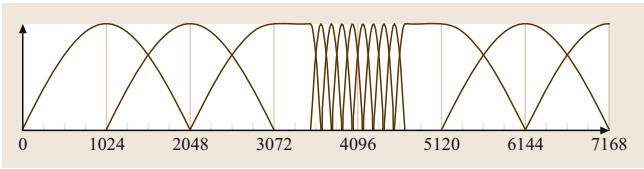


Fig. 18.3 Typical window sequence for AAC block switching (after [18.19], with permission by the AES)

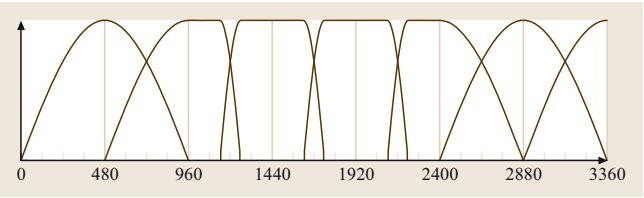


Fig. 18.4 Typical window sequence for AAC-LD window shape adaptation (after [18.19], with permission by the AES)

ing parametric coding of noise components (perceptual noise substitution [18.20]) and long term prediction (LTP). Naturally, all capabilities for handling stereo signals and flexible sampling rates were inherited from MPEG-2 AAC to AAC-LD.

The following modifications were applied to the standard algorithm to achieve low-delay operation:

Frame Length and Filterbank Delay

The frame length has been reduced by a factor of approximately two to 512 or 480 samples, the latter providing a frame length of 10 ms at a sampling rate of 48 kHz, which is commensurate with the granularity of many speech transmission systems. Accordingly, the length of the MDCT analysis window has been reduced to 1024 or 960 time domain samples (corresponding to 512 and 480 spectral values). This leads to an overall framing and filterbank delay of 1024 or 960 samples.

Block Switching

Due to the considerable contribution of the look-ahead time to the encoding delay, block switching is not used. Instead, the temporal spread of quantization noise (*pre-echo*) is handled by the temporal noise shaping (TNS) [18.8] tool.

Window Shape

Besides the standard sine window shape, AAC-LD uses a second window shape which exhibits a lower overlap between subsequent frames. Selection of this window shape for transient signal portions allows the TNS module to provide better protection against pre-echo effects by minimizing the temporal aliasing [18.19] which is inherent in the MDCT's time-domain aliasing cancelation (TDAC) concept [18.3]. Figures 18.3 and 18.4 show a comparison of the typical window sequences of regular AAC and AAC-LD. Note that, in contrast to block switching, this dynamic adaptation of the window shape does not imply any additional encoding delay.

Algorithmic Transmission Delay

Use of the bit reservoir is minimized in order to reach the desired target delay. As an extreme case, no bit reservoir is used at all.

Overall Delay

Taking into account the delay optimizations discussed previously, the AAC-LD overall delay reduces to the following expression:

$$t_{\text{delay}} = \frac{N_{\text{MDCT}} + N_{\text{reduced-bitres}} + N_{\text{frame}}}{F_{\text{s}}} \qquad (18.6)$$

or for packet-based transmission (no bit reservoir):

$$t_{\text{delay}} = \frac{N_{\text{MDCT}}}{F_{\text{s}}} \,. \qquad (18.7)$$

Table 18.1 contrasts this with the delay incurred by a regular AAC encoder/decoder chain. For a window length of 960 samples, a sampling frequency of 48 kHz and packet based transmission (without use of a bit reservoir), an overall algorithmic delay of 20 ms is achieved comparable with common speech codecs (e.g., the GSM full-rate codec).

Table 18.1 Comparison of delay contributions between MPEG-2 AAC and MPEG-4 AAC-LD (expressed in samples)

Delay Source	MPEG-2 AAC	MPEG-4 AAC-LD
Filterbank/framing delay N_{MDCT}	2048	960/1024
Look-ahead delay N_{ahead}	$1024 \cdot \frac{9}{16}$	0
Bit reservoir delay N_{bitres} (assuming a constant rate channel)	up to $6144 \frac{F_{\text{s}}}{\text{bitrate}} - 1024$	As low as 0
Frame transmission delay N_{frame} (assuming a constant rate channel)	1024	480/512

Note that it would be difficult to achieve similar low algorithmic delay values by using MPEG-1 layer 3, mainly due to two reasons: firstly, the composite hybrid filterbank used in layer 3 exhibits a higher filterbank related delay than a plain MDCT with the same spectral resolution. Furthermore, since no TNS tool is available with layer 3, it is necessary to resort to block switching techniques (and accept the associated look-ahead delay) in order to avoid pre-echo problems for transient signals.

A comparison between regular AAC and AAC-LD for two operating points (24 kbit/s AAC versus 32 kbit/s AAC-LD and 54 kbit/s AAC versus 64 kbit/s AAC-LD) showed that the significant decrease in coding delay is achieved at a moderate decrease in coding efficiency by 8 kbit/s per channel [18.19] and that the efficiency of AAC-LD compares favorably with that of an MPEG-1 layer 3 coder.

18.4.3 ITU-T G.722.1-C

The G.722.1 annex C codec, also known under its nickname *Siren 14*, is the 14 kHz audio bandwidth extension of the 7 kHz codec G.722.1 annex A [18.21]. The codec can operate at three possible fixed bit rates of 24, 32, or 48 kbit/s at a fixed sampling rate of 32 kHz. It features an algorithmic delay of 40 ms and is mainly used in video conferencing systems. Figure 18.5 provides an overview of the Siren 14 encoder. While the codec is in general similar to MPEG-4 ER AAC-LD in its basic structure, both codecs differ substantial in their philosophy. An informative synopsis of both schemes is instructive and will be provided in the following.

Both codecs use a cosine modulated filter bank with 50% overlap between subsequent frames which is called modulated lapped transform (MLT) for the Siren codec rather than MDCT.

Both codecs split the spectrum into frequency bands to allow a different quantization step size for spectral components in each band. G.722.1 features frequency bands of constant width (500 Hz), called *regions,* in contrast to AAC-LD (and AAC in general) that groups spectral lines into nonuniform bands that relate to the perceptual frequency scales (e.g., BARK, ERB) of the human auditory system. Effectively, the non-uniform grouping allows more-precise shaping of the introduced quantization noise at lower frequencies (and a coarser control at higher frequencies).

Both Siren 14 and AAC-LD also make use of similar Huffman coding techniques for the entropy coding of the quantized spectral coefficients. The selection procedure of the most suitable Huffman codebook is, however, based on different approaches: whereas the Siren codebook choice is strictly derived from the energy of the audio signal (which is transmitted as side information

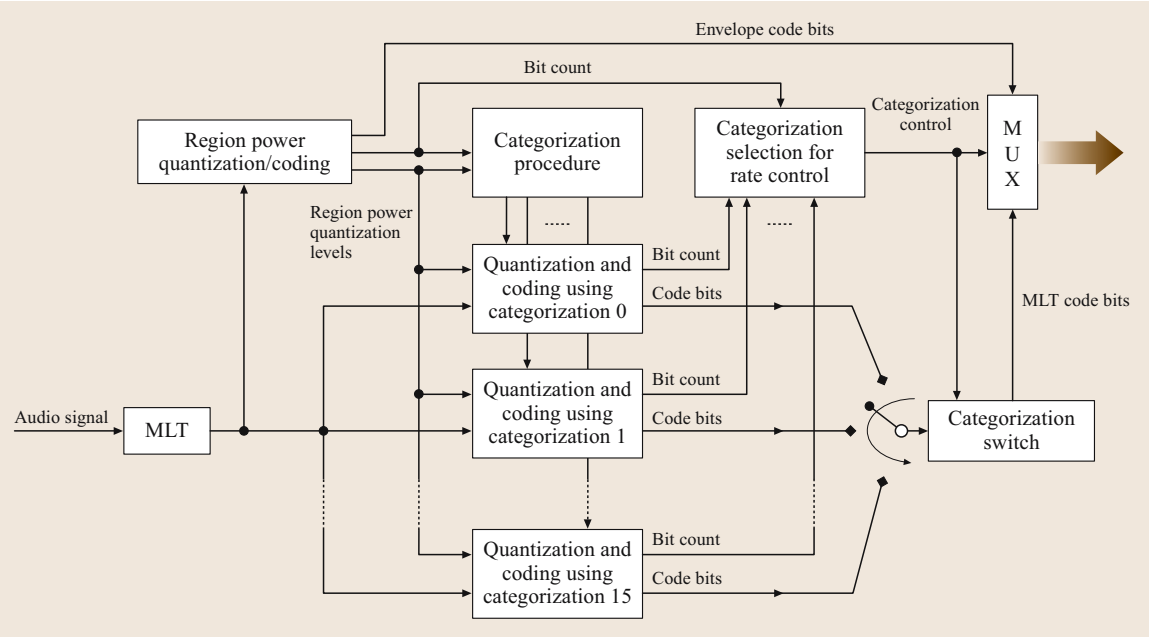


Fig. 18.5 Structure of a G.722.1 encoder (after [18.21])

to the decoder), the AAC encoder can freely select the proper codebooks (usually according to the minimum bit demand) and explicitly conveys its choice to the decoder.

The most obvious difference between both systems is the absence of an explicit psychoacoustic model in the Siren 14 codec as compared to the common model of perceptual coding (see Fig. 18.1). Instead, the decision of the quantization step size (here called *categorization*), choice of Huffman codebook and other parameters are completely derived from the region power information which is transmitted to the decoder as side information. As the only further encoding choice, the Siren 14 encoder selects one out of 16 possible categorizations (*categorization selection for rate control*) and transmits this selection as four *categorization control bits* to the decoder to accommodate a constant transmission bit rate.

Many of the design differences between AAC-LD and Siren 14 originate from the different philosophies of the associated standardization bodies. While ITU-T standardizes bit-exact audio encoders and decoders to guarantee a known and reliable signal quality, MPEG restricts standardization to the bitstream format and decoder behavior to ensure interoperability and, at the same time, allow further improvements in encoder implementation and audio quality over time. As a consequence, MPEG codecs traditionally leave many decisions that influence the resulting audio quality to the encoder algorithm and transmit these decisions explicitly in the bitstream. In contrast, an ITU-T codec implements a fixed overall algorithm in a maximally efficient way with regard to transmission of side information, frequently deriving decisions implicitly from other known parameters such as band energies. The advantage of this approach is a low side information rate which comes at the price of restricted algorithmic flexibility. In summary, Siren 14 is designed as a simpler system than AAC-LD in terms of algorithmic structure (including coding tools) and computational complexity. Due to its compact side information structure it provides good audio quality for average music and speech signals at low bit rates. On the other side, it exhibits weakness with critical audio material such as transient signals, and does not benefit from a higher bit rate to the same extent known from AAC codecs [18.22].

18.4.4 Ultra-Low-Delay Perceptual Audio Coding

Traditionally, speech coding is based on a predictive structure which provides a very low encoding/decoding delay. In contrast, perceptual audio coding requires the presence of a psychoacoustic model plus the ability to control the spectral shape of the quantization distortion to meet the masking requirements. Both requirements call for a subband spectral decomposition of the audio signal. Hence, traditional perceptual modeling and coding can not be applied directly to predictive coding of a time domain signal. To solve this problem, the ultra-low-delay (ULD) codec [18.23, 24] makes use of a *pre-/postfiltering* approach [18.25] that separates the application of psychoacoustics (i. e., the irrelevance reduction) from the redundancy reduction, so that two separate processing steps are applied.

Figure 18.6 shows the general structure of the ULD encoder and decoder.

Very much like in traditional filterbank-based perceptual audio coding, the input of the ULD perceptual model computes subband coefficients from an analysis filterbank. For the purpose of this coder, however, a rather small number of subbands are chosen providing both sufficient frequency and time resolution to model

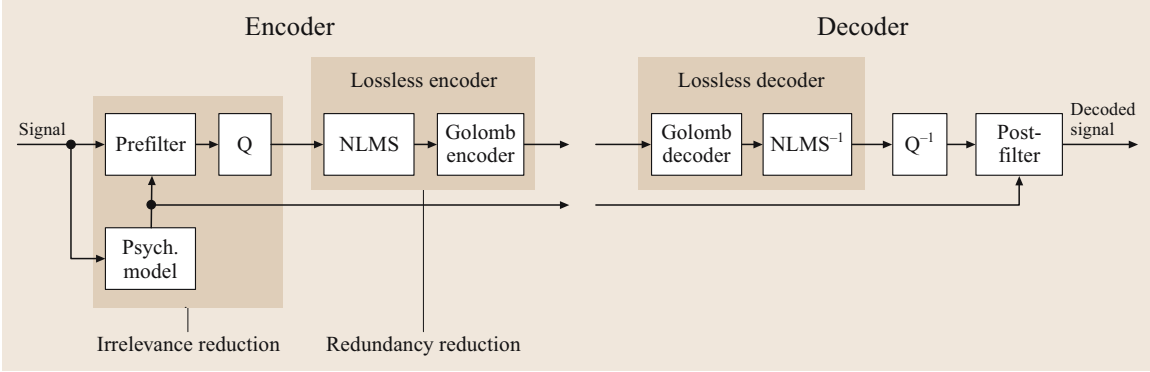


Fig. 18.6 Structure of ultra-low-delay (ULD) audio encoder/decoder

spectral and temporal masking effects. The output of the perceptual model is the masking threshold for each subband. The input signal passes through the prefilter, i. e., a time-varying linear filter the coefficients of which are continuously updated such that its magnitude response is the inverse of the masking threshold. As a consequence, it *normalizes* the signal with regard to its masking threshold. The filter coefficients for the postfilter are transmitted as side information to the decoder. Since the prefiltered signal contains all audio spectral components weighted according to their perceptual relevance, it is subsequently quantized by a simple time-domain signal quantizer (e.g., a scalar uniform quantizer), producing a quantization error that is flat over frequency. In this way, the combination of perceptual model, prefilter and quantizer exploits the *irrelevance* of the input signal. The delay of this stage is determined by the perceptual model and its filterbank. For an implementation using a filterbank with 128 uniformly spaced subbands and a window length of 256, this leads to a delay of 128 samples. Additionally, the interpolation of the masking threshold values between subsequent blocks introduces another 128 samples, totalling to 256 samples. The quantized time domain values produced by the irrelevance reduction stage are then further compressed by a subsequent *redundancy reduction* module which aims at lossless encoding of its input signal with a minimum number of bits using both predictive coding techniques and entropy coding.

Common lossless audio codecs are typically based on blockwise forward adaptive prediction (i. e., the prediction coefficients for a block are transmitted as side information), implying a delay of at least one block size. To avoid this additional delay, backward adaptive predictive coding can be used as it is known from low-delay speech [18.26] or audio coding [18.27]. While these coders employ backward adaptive prediction to update LPC filters, the ULD coder makes use of this technique to obtain a lossless compression of quantized (integer) values. The backward adaptation is implemented by using the popular normalized least-mean-square (NLMS) algorithm [18.28]. The NLMS predictor is followed by an entropy coding with low delay, such as arithmetic coding, adaptive Huffman coding or Golomb coding. The latter has the advantage of a low computational complexity and no additional delay. Constant bit rates can be achieved by an encoder rate loop including quantizer, prediction module, and entropy encoder [18.29].

On the decoder side, the lossless decoder performs the inverse entropy and predictive coding steps, and the quantized values are then mapped back to their original scaling. Finally, the audio output signal is produced by applying a postfilter which operates as the inverse counterpart of the prefilter. Hence, this filter undoes the perceptual normalization of the input signal introduced by the prefilter. By imposing a magnitude response resembling the masking threshold curve, it effectively also shapes the spectrally flat quantization noise to match this curve.

The overall algorithmic encoding/decoding delay of a typical ULD codec is about 256 samples corresponding to about 8 ms at 32 kHz sampling rate or below 6 ms at 48 kHz, possible bit rates ranging from around 32 to 96 kbit/s per channel, where high audio quality can be achieved at 80 kbit/s per channel. The flexibility in block length and psychoacoustic look-ahead can be used to derive even more delay optimized versions. As an example, the delay can be reduced to 1.5 ms at a cost of around 1/3 in bit rate. The robustness of the ULD decoder against transmission errors can be significantly improved by applying an adaptive reshaping of the prediction error [18.30]. A ULD encoder consumes less than 100 MHz on a 16 bit fixed-point digital signal processor (DSP, Analog Devices BF533) in a configuration with 48 kHz sampling rate, constant bit rate, 64-tap NLMS predictor and Golomb encoding.

18.5 Hybrid/Crossover Coders

Perceptual audio coding has made significant advances in performance over time. Nonetheless, the ability of using a source model, as it is used by speech coders, provides a distinct advantage for coding speech signals over generic perceptual audio coding at very low bit rates (e.g., below 20 kbit/s per channel). Typically, when running at very low bit rates, filterbank-based perceptual audio coders tend to smear the temporal structure of individual glottal excitation pulses as a consequence of coarsely quantizing spectral coefficients. This often leads to a *reverberant* quality of the decoded signal.

Conversely, attempting to use traditional speech coders for sound that is not a pure speech signal (e.g., speech with background noise or multiple talkers) re-

sults in significant distortion because the input signal does not conform to the speech production model embedded into the coder. Moreover, coding of complex signals, such as music, with speech coders usually produces extremely undesirable artifacts due to the coder's inability to accurately reproduce tonal signal components with a frequency outside the pitch range of human voice.

These complementary characteristics of speech coders and perceptual audio coders have repeatedly motivated researchers to look into combining both schemes into a single architecture such that the strengths of both approaches are preserved. The following sections will discuss a number of such approaches as they have emerged over time.

18.5.1 MPEG-4 Scalable Speech/Audio Coding

One of the first combinations between speech coding technology and perceptual audio coding was conceived as part of the scalable MPEG-4 audio coder [18.13]. It encodes audio and speech material in a way that allows to adapt the bit rate in response to the time-varying channel capacity of transmission networks without the need for reencoding with the instantaneous target bit rate. Among other scalable codec combinations, MPEG-4 supports a hybrid scheme including both an MPEG-4 speech coder and (or several) AAC audio coders.

In this configuration, a scalable bitstream consists of a CELP core layer bitstream and one or more AAC enhancement layers. The core bitstream can be decoded on its own, resulting in a meaningful decoder output with normal speech coding quality and restrictions. Decoding of both the core bitstream and the AAC enhancement layers will increase output quality and bandwidth substantially over the core layer. Thus, transmission or decoding of a subset of the full bitstream will lead to a valid decoded signal though at a lower quality. The structure of such a coder is depicted by Fig. 18.7. The CELP base layer coder operates at a lower sampling rate than the subsequent enhancement layers. The scalable combination of these component coder works as follows:

- The audio input is downsampled and encoded by the CELP coder. The produced bitstream constitutes the base layer portion of the scalable bitstream. It is decoded locally and upsampled to match the sampling rate of the T/F-based enhancement layers and passed through an MDCT analysis filterbank. Since the ratio of core to enhancement layer sampling rate is normally an integer value, the upsampling operation can be implemented simply, e.g., by inserting zeros values and band limiting of the spectral coefficients after the filterbank.
- In a parallel signal path, the delay compensated input signal is passed through the MDCT analysis filterbank, and the residual coding error signal is computed.
- The residual signal is passed through a frequency selective switch (FSS) which permits to fall back to the original signal on a scale-factor band basis if this can be coded more efficiently.
- The resulting spectral coefficients are quantized/coded by an AAC coding kernel, leading to an enhancement layer bitstream.

Figure 18.8 shows the corresponding decoder. The core layer is decoded and upsampled to match the sampling rate of the enhancement layer and converted into an MDCT representation. This spectral representation is then refined by combining it with the spectral data decoded from the enhancement layer using an inverse frequency selective switch (IFSS). Finally, an inverse MDCT (IMDCT) converts the signal back into a time-domain output.

Further refinement stages (enhancement layers) could be added by continued recoding of the residual coding error signal, and can be computed efficiently in the spectral domain. Since each enhancement layer car-

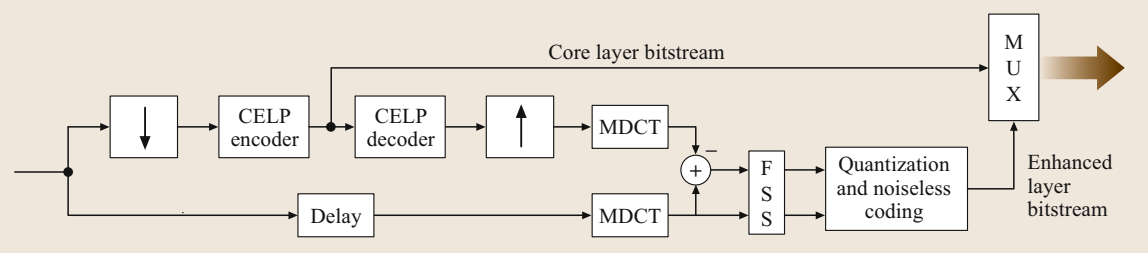


Fig. 18.7 Scalable MPEG-4 coder with speech coding core and general audio enhancement

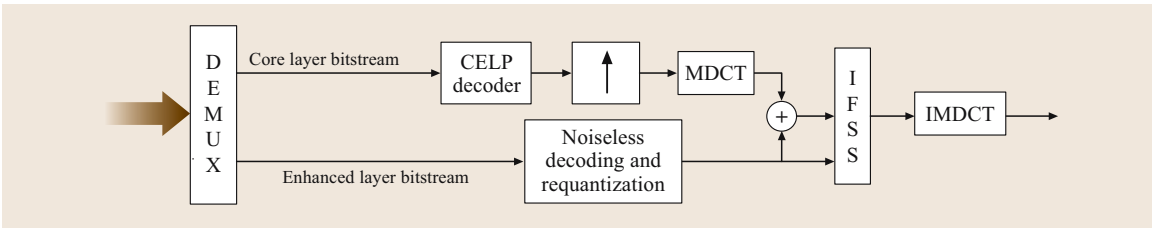


Fig. 18.8 Scalable MPEG-4 decoder with speech coding core and general audio enhancement

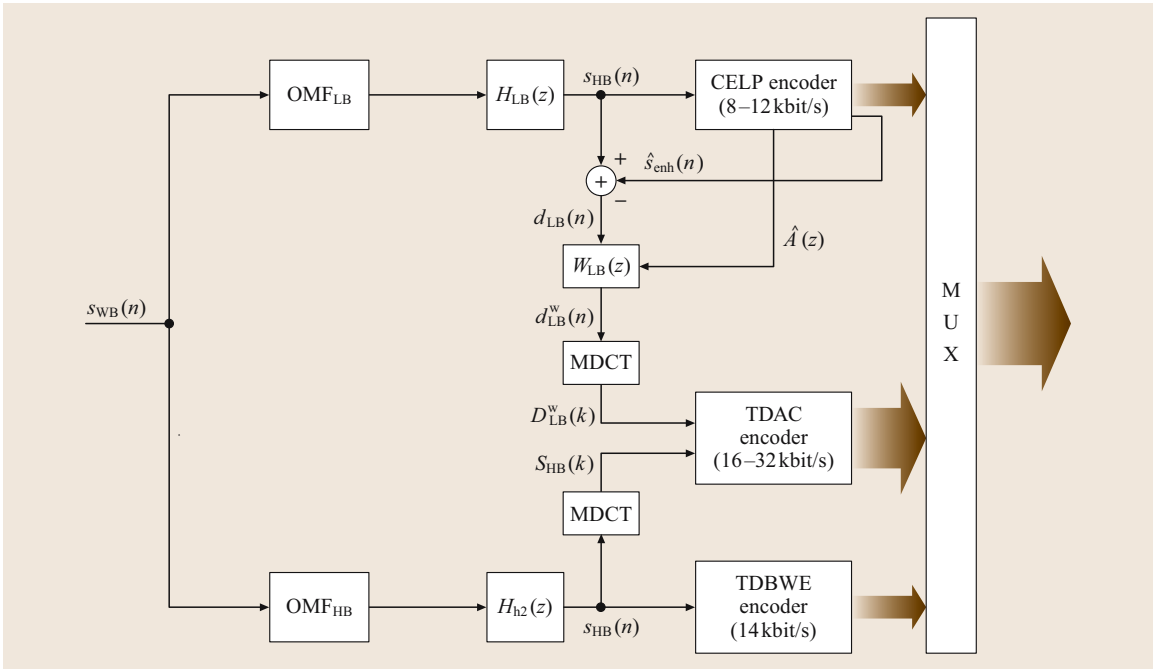


Fig. 18.9 Structure of a G.729.1 encoder (after [18.31])

ries some associated side information overhead (e.g., scale-factor data), this type of scalability is efficient for sufficiently large enhancement steps (typically at least 8 kbit/s), and is therefore referred to as *large step scalability* in MPEG-4 Audio. A more-extensive discussion MPEG-4 audio scalability features is provided in [18.13].

18.5.2 ITU-T G.729.1

Another example of a coder that combines CELP speech coding with a filterbank-based enhancement stage is the recent ITU-T G.729.1 scheme. It offers a 12-layer scalable (*embedded*) bitstream structure with bit rates between 8 kbit/s and 32 kbit/s, and is interoperable with widely deployed G.729 systems. For 8 or 12 kbit/s, the coder provides traditional narrow-band coding capability, and at higher bit rates it is scalable in steps of 2 kbit/s, transmitting wide-band signals (50 Hz–7000 Hz).

Figure 18.9 illustrates the G.729.1 encoder architecture. In order to accommodate both narrow- and wide-band operation, the 16 kHz sampling rate input signal $s_{WB}(n)$ is split into decimated lower-frequency half-band $s_{LB}(n)$ and decimated upper-frequency half-band $s_{HB}(n)$ signals by a quadrature mirror filter (QMF). These signals are then coded by three stages:

- The backward-compatible base codec consists of a narrow-band CELP core operating at 8 kbit/s (layer 1 only) or 12 kbit/s (layers 1 and 2) on a 50 Hz high-pass-filtered version of the lower-frequency band signal $s_{LB}(n)$.

- The CELP coding error (residual signal) $d_{LB}(n)$ is perceptually filtered by $W_{LB}(z)$ (whose parameters are derived from the LPC module of the CELP encoder) and transformed into the MDCT domain. This perceptually weighted difference signal D^{w}_{LB} is combined with the MDCT-domain high-band signal $S_{HB}(k)$ in the time-domain aliasing cancelation codec and transmitted within layers 4–12, resulting in bit rates from 16 kbit/s up to 32 kbit/s.
- In order to enable a full wide-band representation of the audio signal for bit rates at or above 14 kbit/s, layer 3 adds information for a parametric time-domain bandwidth extension (TDBWE) tool in the decoder.

Figure 18.10 shows the structure of the corresponding decoder. Layers 1 and, if present, 2 are decoded by the embedded CELP decoder, postfiltered and post-processed by a high-pass filter (HPF) to obtain the low-frequency input signal for the QMF synthesis filterbank (IQMF). If layer 3 is present (i. e., for bit rates larger or equal to 14 kbit/s), the TDBWE decoder produces a high-frequency synthesis which is further processed in the MDCT domain, and transformed back to provide the high-frequency band signal for the QMF synthesis filterbank. For all further layers, the TDAC decoder produces MDCT coefficients which describe the (weighted) residual signal for the lower-frequency QMF band or the reconstructed signal in the high-frequency QMF band. In the latter case, the transmitted MDCT coefficients replace the TDBWE signal as they become available. Both the low- and high-frequency TDAC decoder contributions are passed through an inverse MDCT. In order to reduce pre/post-echo artifacts (i. e., temporal smearing of the signal resulting from processing with a high frequency resolution), such conditions are detected and the corresponding signals are modified based on the time envelope information from the CELP and TDBWE parts.

The overall algorithmic delay of the codec is 48.94 ms, and its complexity is around 35 wmops ([18.31, Table 5]).

18.5.3 AMR-WB+

Another interesting and recent example of a hybrid speech/audio coding scheme is the extended wide-band adaptive multirate coder (AMR-WB+) [18.32] which extends the wide-band AMR speech coder (AMR-WB) towards the ability of handling general audio signals. As an important difference from the coders discussed in the preceding two sections (MPEG-4 Scalable Audio Coding in Sect. 18.5.1 and G.729.1 in Sect. 18.5.2), this

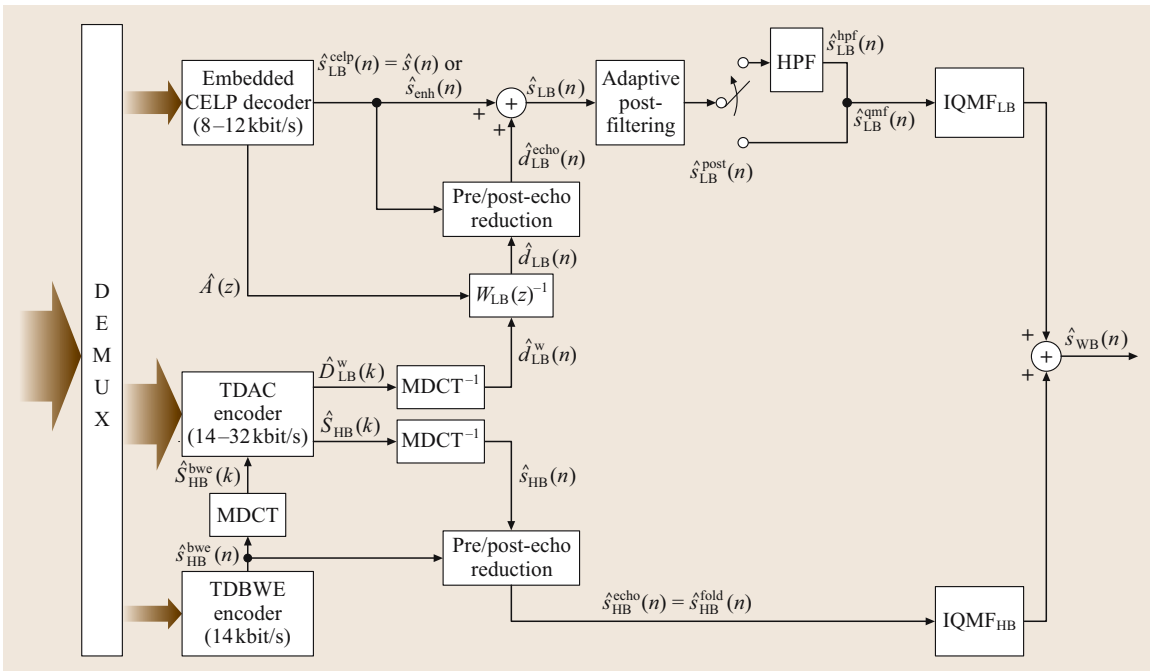


Fig. 18.10 Structure of a G.729.1 decoder (after [18.31])

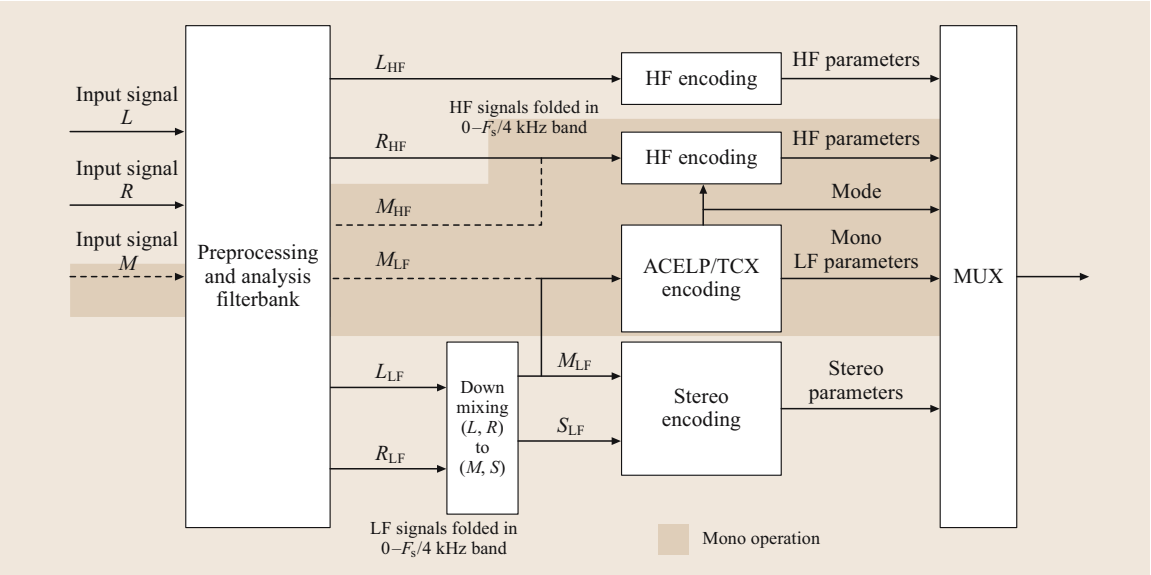


Fig. 18.11 High-level structure of the extended AMR-WB encoder (after [18.32])

flexibility is not achieved by running a filterbank-based audio coding on top of (i. e., as an enhancement layer to) an underlying speech coder, but by allowing a *dynamic selection* between two alternate coding modes. While the algebraic code excited linear prediction (ACELP) mode handles speech-like signals in an efficient way, a filterbank-based audio coding mode (transform coded excitation, TCX) provides good quality for non-speech signal types. As a further extension, the AMR-WB+ coder includes a parametric stereo processing for efficient handling of stereo audio signals.

Figure 18.11 shows the high-level structure of the encoder. The input signal is first converted to the coder's internal sampling frequency (nominally 25.6 kHz), and then split into two equal-sized subbands (denoted with the subscripts *HF* and *LF* for high- and low-frequency band sin the figure, respectively). For operation with a mono input signal (denoted *M* in the figure), only the parts with the grey background are used. Each block of 1024 subband samples (called a superframe) is processed independently. The lower subband signal (nominally between 0 and 6.4 kHz) is encoded using the hybrid ACELP/TCX coder while the high band (nominally between 6.4 and 12.8 kHz) is encoded using a bandwidth extension method.

If a stereo signal is encoded (signals *L* and *R* in the figure), its high frequency subband signals are encoded separately using the bandwidth extension mechanism. The low-frequency band of the stereo signal is converted into *mid* (sum) and *side* (difference) signals, *M* and *S*, respectively. While the mid signal is fed into the ACELP/TCX coder, the side signal is further processed by the stereo encoding stage (i. e., parametrically coded above 1 kHz nominal, and ACELP/TCX coded below this frequency). All quantized and coded parameters are multiplexed into a bitstream.

At the heart of the coding scheme, the ACELP/TCX core divides each superframe of 1024 samples into four frames of 256 samples, each of which can be encoded in four possible modes:

- as a 256-sample ACELP frame
- as a 256-sample TCX frame
- as a part of a larger 512-sample TCX frame that is constructed by concatenating two subsequent frames
- as a part of a long 1024-sample TXC frame that encompasses the whole superframe

The selection of the optimum coding mode is carried out on a per-frame basis and can either be done in a closed- or open-loop fashion. The latter is computationally less demanding but results in some quality degradation for non-speech signals. A more-detailed description of the hybrid ACELP/TCX coding approach can be found in [18.33].

The AMR-WB+ decoder (Fig. 18.12) demultiplexes the received bitstream, recovers the quantized parameters and performs the corresponding synthesis operations to reconstruct the output signal.

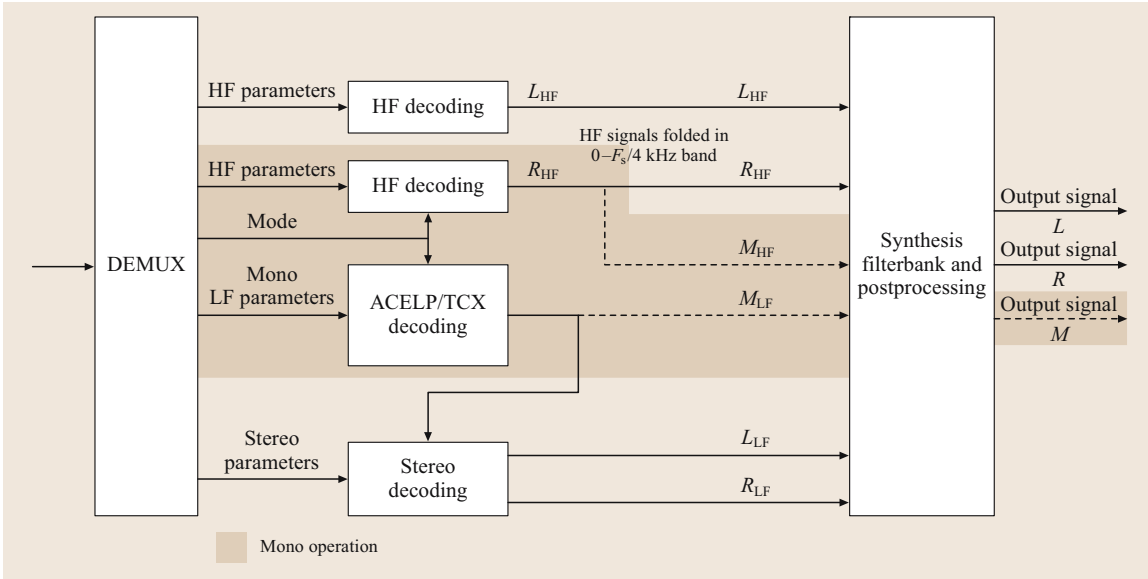


Fig. 18.12 High-level structure of the extended AMR-WB decoder (after [18.32])

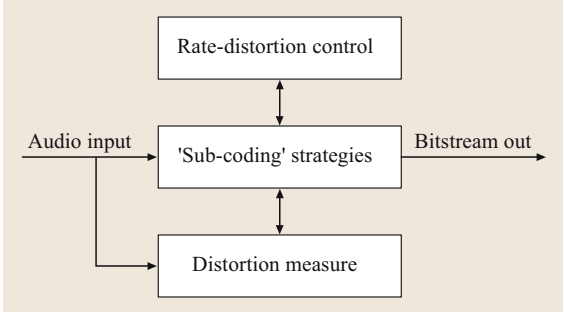


Fig. 18.13 High-level structure of the ARDOR coder (after [18.36])

The AMR-WB+ codec supports various sampling frequencies between 8 and 48 kHz and bit rates of 6–36 kbit/s (mono operation)/8–48 kbit/s (stereo operation). When running at the nominal internal sampling frequency, the codec features an overall algorithmic delay of about 113.8 ms [18.34]. For a more-detailed overview of AMR-WB+ see [18.35].

18.5.4 ARDOR

While all of the combined speech/audio coders discussed previously are the results of intense development work within standardization bodies, there is also considerable activity in the research community on the topic of hybrid coding. One example is the coder that was developed under the *adaptive rate-distortion optimized sound coder* (ARDOR) European research project [18.36] and aims at providing a unified universal audio coder that shows optimal behavior for a wide range of bit rates, input signal types and other requirements (such as delay). The general architecture of the coder is shown in Fig. 18.13. The key idea is to process the input signal using a number of *subcoding* strategies (i. e., coding modules such as CELP-based speech coding, filterbank-based audio coding and sinusoidal audio coding). This may happen either individually (i. e., one subcoder at a time) or, more generally, in combination (i. e., each subcoder carries a part of the signal to be coded). In order to make optimum use of the pool of subcoders, a central element of the concept lies in the *rate-distortion (R/D) optimization* approach which helps select the subcoder that provides the best performance in a rate-distortion sense for a given input signal (or parts thereof). This gives the framework for an overall optimization of the whole system. As a third important aspect of the overall system, the rate-distortion optimization process employs a *perceptual distortion measure* in order to optimize subjective quality rather than an inappropriate simple metric like SNR. Thus, accurate modeling of perceptual quality metrics is paramount to guiding the overall system in a meaningful way. An example of R/D optimization in the ARDOR context can be found in [18.37].

Naturally, a coder resulting from such a research project cannot compete with highly optimized standardized schemes in the marketplace in terms of its implementation and computational complexity. Instead, it delivers a proof of feasibility for a comprehensive approach to the universal coding problem and provides valuable insights into the underlying problems. Finally, it demonstrates the possible gain attainable from such a concept, and in this way further pushes the borders. More work in this area is needed to arrive at a practically attractive solution.

18.6 Summary

Perceptual audio coding has made significant progress during the last two decades and is becoming increasingly attractive not only for music signals but also for carrying high-quality speech signals. A number of perceptual audio coding schemes have been developed fulfilling the delay requirements imposed by real-time two-way voice communication and thus enabling telephone and teleconferencing applications with high sound reproduction fidelity. While perceptual audio coders cannot offer the same speech compression efficiency as their source-model-based speech coding counterparts at very low bit rates, they traditionally excel in terms of quality that scales continuously with bit rate without saturating and their tolerance to mixed and complex signals.

It is interesting to see the first signs of successful convergence between the speech and the audio coding worlds in a quest for hybrid schemes that combine the advantages of both coding paradigms into a single system on the way towards the *universal coder*.

References

18.1 B.C.J. Moore: *Introduction to the Psychology of Hearing*, 3rd edn. (Academic, New York 1989)

18.2 E. Zwicker, H. Fastl: *Psychoacoustics, Facts and Models* (Springer, Berlin, Heidelberg 1990)

18.3 J. Princen, A. Johnson, A. Bradley: Subband/Transform Coding Using Filter Bank Designs Based on Time Domain Aliasing Cancellation, IEEE ICASSP, 2161–2164 (1987)

18.4 J.H. Rothweiler: Polyphase Quadrature Filters – a new Subband Coding Technique, IEEE ICASSP, 1280–1283 (1983)

18.5 K. Brandenburg, E. Eberlein, J. Herre, B. Edler: Comparison of Filterbanks for High Quality Audio Coding, IEEE ISCAS (1992)

18.6 M. Bosi: Filter Banks in Perceptual Audio Coding, Proc. of the 17th International AES Conference on High Quality Audio Coding (1999)

18.7 R.P. Hellman: Asymmetry of Masking between Noise and Tone, Percept. Psychophys. **11**, 241–246 (1972)

18.8 J. Herre: Temporal Noise Shaping, Quantization and Coding Methods in Perceptual Audio Coding: A Tutorial Introduction, Proc. of the 17th International AES Conference on High Quality Audio Coding (1999)

18.9 ISO/IEC: JTC1/SC29/WG11 MPEG International Standard ISO/IEC 11172, Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s (ISO, Geneva 1993)

18.10 ISO/IEC: JTC1/SC29/WG11 MPEG International Standard ISO/IEC 13818-3, Generic Coding of Moving Pictures and Associated Audio: Audio (1994)

18.11 ISO/IEC: JTC1/SC29/WG11 MPEG International Standard ISO/IEC 13818-7, Generic Coding of Moving Pictures and Associated Audio: Advanced Audio Coding (1997)

18.12 ISO/IEC: JTC1/SC29/WG11 MPEG International Standard ISO/IEC 14496-3:2001, Coding of Audio-Visual Objects, Part 3 Audio (2001)

18.13 F. Pereira, T. Ebrahimi (Eds.): *The MPEG-4 Book, IMSC Multimedia Series* (Prentice Hall, Englewood Cliffs 2002)

18.14 ISO/IEC: JTC1/SC29/WG11 MPEG 14496-3:2001/Amd.1:2003, Coding of Audio-Visual Objects – Part 3: Audio, Amendment 1: Bandwidth extension (2003)

18.15 M. Dietz, L. Liljeryd, K. Kjoerling, O. Kunz: *Spectral Band Replication, a Novel Approach in Audio Coding* (112th AES Convention, Munich 2002), Preprint 5553

18.16 ISO/IEC: JTC1/SC29/WG11 MPEG 14496-3:2001/Amd.1:2003, Coding of Audio-Visual Objects – Part 3: Audio, Amendment 2: Parametric coding for high quality audio (2004)

18.17 W. Oomen, E. Schuijers, B. den Brinker, J. Breebaart: *Advances in Parametric Coding for High-Quality Audio* (114th AES Convention, Amsterdam 2002), Preprint 5852

18.18 B. Edler: Codierung von Audiosignalen mit überlappender Transformation und adaptiven Fensterfunktionen, Frequenz **43**, 252–256 (1989), in German

18.19 E. Allamanche, R. Geiger, J. Herre, T. Sporer: *MPEG-4 Low Delay Audio Coding based on the AAC Codec* (106th AES Convention, Munich 1999), Preprint 4929

18.20 J. Herre, D. Schulz: *Extending the MPEG-4 AAC Codec by Perceptual Noise Substitution* (104th AES Convention, Amsterdam 1998), Preprint 4720

18.21 ITU-T Recommendation G.722.1 (5/2005): Low-complexity coding at 24 and 32 kbit/s for hands-free operation in systems with low frame loss

18.22 R. Geiger, M. Lutzky, M. Schmidt, M. Schnell: *Structural Analysis of Low Latency Audio Coding Scheme* (119th AES Convention, New York 2005), Preprint 6601

18.23 G. Schuller, A. Härmä: Low Delay Audio Compression using Predictive Coding, IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Orlando (2002)

18.24 G. Schuller, B. Yu, D. Huang, and B. Edler: Perceptual Audio Coding using Adaptive Pre and Post-Filters and Lossless Compression, IEEE Transactions on Speech and Audio Processing (2002) pp. 379–390

18.25 B. Edler, G. Schuller: Audio Coding Using a Psychoacoustic Pre- and Post-Filter, IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Istanbul (2000)

18.26 J.-H. Chen, R.V. Cox, Y.-C. Lin, N. Jayant, M.J. Melchner: A low-delay CELP coder for the CCITT 16 kb/s speech coding standard, IEEE J. Sel Areas in Comm **10**, 830–849 (1992)

18.27 A. Härmä, U. K. Laine, and M. Karjalainen: Backward adaptive warped lattice for wideband stereo coding in Proc. of EUSIPCO '98, Greece (1998)

18.28 S.S. Haykin: *Adaptive Filter Theory* (Prentice Hall, Englewood Cliffs 1999)

18.29 U. Krämer, G. Schuller, S. Wabnik, J. Klier, J. Hirschfeld: Ultra Low Delay audio coding with constant bit rate, 117th AES Convention, San Francisco, Preprint 6197

18.30 S. Wabnik, G. Schuller, J. Hirschfeld, U. Kraemer: Packet Loss Concealment in Predictive Audio Coding, IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, (2005) New Paltz

18.31 ITU-T Recommendation G.729.1 (5/2006): G.729 based Embedded Variable bit-rate coder: An 8–32 kbit/s scalable wideband coder bitstream interoperable with G.729

18.32 GSM 3rd Generation Partnership Project (3GPP), 3GPP TS 26.290: Audio codec processing functions; Extended AMR Wideband codec; Transcoding functions

18.33 B. Bessette, R. Lefebvre, and R. Salami: Universal Speech/Audio Coding Using Hybrid ACELP/TCX Techniques, IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Philadelphia (2005)

18.34 ETSI TR 126 936 V6.1.0 (2006-03): Universal Mobile Telecommunications System (UMTS), Performance characterization of 3GPP audio codecs

18.35 R. Salami, R. Lefebvre, K. Kontola, S. Bruhn, A. Taleb: Extended AMR-WB for high-quality audio on mobile devices, IEEE Commun. Mag. **44**(5), 90–97 (2006)

18.36 N.H. van Schijndel, S. van de Par: Rate-distortion optimized hybrid sound coding. In:, Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA 2005) (2005) pp. 235–238

18.37 R. Vafin, W. B. Kleijn: Rate-Distortion Optimized Quantization in Multistage Audio Coding, IEEE Transactions on Speech and Audio Processing 14:311-320 (2006)

14. Principles of Speech Coding

W. B. Kleijn

Speech coding is the art of reducing the bit rate required to describe a speech signal. In this chapter, we discuss the attributes of speech coders as well as the underlying principles that determine their behavior and their architecture. The ubiquitous class of linear-prediction-based coders is used as an illustration. Speech is generally modeled as a sequence of stationary signal segments, each having unique statistics. Segments are encoded using a two-step procedure: (1) find a model describing the speech segment, (2) encode the segment assuming it is generated by the model. We show that the bit allocation for the model (the predictor parameters) is independent of overall rate and of perception, which is consistent with existing experimental results. The modeling of perception is an important aspect of efficient coding and we discuss how various perceptual distortion measures can be integrated into speech coders.

14.1 The Objective of Speech Coding

In modern communication systems, speech is represented by a sequence of bits. The main advantage of this *binary* representation is that it can be recovered exactly (without distortion) from a noisy channel (assuming proper system design), and does not suffer from decreasing quality when transmitted over many transmission legs. In contrast, analog transmission generally results in an increase of distortion with the number of legs.

An acoustic speech signal is inherently analog. Generally, the resulting analog microphone output is converted to a binary representation in a manner consistent with Shannon's sampling theorem. That is, the analog signal is first band-limited using an anti-aliasing filter, and then simultaneously sampled and quantized. The output of the *analog-to-digital (A/D) converter* is a digital speech signal that consists of a sequence of numbers of finite precision, each representing a sample of the band-limited speech signal. Common sampling rates are 8 and 16 kHz, rendering *narrowband speech* and *wideband speech*, respectively, usually with a precision of 16 bits per sample. For the 8 kHz sampling rate a logarithmic 8-bit-per-sample representation is also common.

Particularly at the time of the introduction of the binary speech representation, the bit rate produced by the A/D converter was too high for practical applications such as cost-effective mobile communications and secure telephony. A search ensued for more-efficient digital representations. Such representations are possible since the digital speech contains *irrelevancy* (the signal is described with a higher precision than is needed) and *redundancy* (the rate can be decreased without affecting precision). The aim was to trade off computational effort at the transmitter and receiver for the bit rate required for the speech representation. Efficient representations generally involve a *model* and a set of *model parameters*, and sometimes a set of coefficients that form the input to the model. The algorithms used to reduce the required rate are called speech-coding algorithms, or *speech codecs*.

The performance of speech codecs can be measured by a set of properties. The fundamental codec attributes are bit rate, speech quality, quality degradation due to channel errors and packet loss, delay, and computational effort. Good performance for one of the attributes generally leads to lower performance for the others. The interplay between the attributes is governed by the fundamental laws of information theory, the properties of the speech signal, limitations in our knowledge, and limitations of the equipment used.

To design a codec, we must know the desired values for its attributes. A common approach to develop a speech codec is to constrain all attributes but one quantitatively. The design objective is then to optimize the remaining attribute (usually quality or rate) subject to these constraints. A common objective is to maximize the average quality over a given set of channel conditions, given the rate, the delay, and the computational effort.

In this chapter, we attempt to discuss speech coding at a generic level and yet provide information useful for practical coder design and analysis. Section 14.2 describes the basic attributes of a speech codec. Section 14.3 discusses the underlying principles of coding and Sect. 14.4 applies these principles to a commonly used family of linear predictive (autoregressive model-based) coders. Section 14.5 discusses distortion criteria and how they affect the architecture of codecs. Section 14.6 provides a summary of the chapter.

14.2 Speech Coder Attributes

The usefulness of a speech coder is determined by its attributes. In this section we describe the most important attributes and the context in which they are relevant in some more detail. The attributes were earlier discussed in [14.1, 2].

14.2.1 Rate

The rate of a speech codec is generally measured as the average number of bits per second. For *fixed-rate* coders the bit rate is the same for each coding block, while for *variable-rate* coders it varies over time.

In traditional circuit-switched communication systems, a fixed rate is available for each communication direction. It is then natural to exploit this rate at all times, which has resulted in a large number of standardized fixed-rate speech codecs. In such coders each particular parameter or variable is encoded with the same number of bits for each block. This a priori knowledge of the bit allocation has a significant effect on the structure of the codec. For example, the mapping of the quantization indices to the transmitted codewords is trivial. In more-flexible circuit-switched networks (e.g., modern mobile-phone networks), codecs may have a variable number of modes, each mode having a different fixed rate [14.3,4]. Such codecs with a set of fixed coding rates should not be confused with true variable-rate coders.

In variable-rate coders, the bit allocation within a particular block for the parameters or variable depends on the signal. The bit allocation for a parameter varies with the quantization index and the mapping from the quantization index to the transmitted codeword is performed by means of a table lookup or computation, which can be very complex. The major benefit of variable-rate coding is that it leads to higher coding efficiency than fixed-rate coders because the rate constraint is less strict.

In general, network design evolves towards the facilitation of variable-rate coders. In packet-switched communication systems, both packet rate and size can vary, which naturally leads to variable-rate codecs. While variable-rate codecs are common for audio and video signals, they are not yet commonplace for speech. The requirements of low rates and delays lead to a small packet payload for speech signals. The relatively large packet header size limits the benefits of the low rate and,

consequently, the benefit of variable-rate speech coding. However, with the removal of the fixed-rate constraint, it is likely that variable-rate speech codecs will become increasingly common.

14.2.2 Quality

To achieve a significant rate reduction, the parameters used to represent the speech signal are generally transmitted at a reduced precision and the reconstructed speech signal is not a perfect copy of the original digital signal. It is therefore important to ensure that its quality meets a certain standard.

In speech coding, we distinguish two applications for quality measures. First, we need to evaluate the *overall quality* of a particular codec. Second, we need a *distortion measure* to decide how to encode each signal block (typically of duration 5–25 ms). The distortion measure is also used during the design of the coder (in the training of its codebooks). Naturally, these quality measures are not unrelated, but in practice their formulation has taken separate paths. Whereas overall quality can be obtained directly from scoring of speech utterances by humans, distortion measures used in coding algorithms have been defined (usually in an ad hoc manner) based on knowledge about the human auditory system.

The only true measure of the overall quality of a speech signal is its rating by humans. Standardized conversational and listening tests have been developed to obtain reliable and repeatable (at least to a certain accuracy) results. For speech coding, listening tests, where a panel of listeners evaluates performance for a given set of utterances, are most common. Commonly used standardized listening tests use either an absolute category rating, where listeners are asked to score an utterance on an absolute scale, or a degradation category rating, where listeners are asked to provide a relative score. The most common overall measure associated with the absolute category rating of speech quality is the mean opinion score (MOS) [14.5]. The MOS is the mean value of a numerical score given to an utterance by a panel of listeners, using a standardized procedure. To reduce the associated cost, subjective measures can be approximated by objective, repeatable algorithms for many practical purposes. Such measures can be helpful in the development of new speech coders. We refer to [14.6–8] and to Chap. 5 for more detail on the subject of overall speech quality.

As a distortion measure for speech segments variants of the squared-error criterion are most commonly used. The squared-error criterion facilitates fast evaluation for coding purposes. Section 14.5 discusses distortion measures in more detail. It is shown that adaptively weighted squared error criteria can be used for a large range of perceptual models.

14.2.3 Robustness to Channel Imperfections

Early terrestrial digital communication networks were generally designed to have very low error rates, obviating the need for measures to correct errors for the transmission of speech. In contrast, bit errors and packet loss are inherent in modern communication infrastructures.

Bit errors are common in wireless networks and are generally addressed by introducing channel codes. While the integration of source and channel codes can result in higher performance, this is not commonly used because it results in reduced modularity. Separate source and channel coding is particularly advantageous when a codec is faced with different network environments; different channel codes can then be used for different network conditions.

In packet networks, the open systems interconnection reference (OSI) model [14.9] provides a separation of various communication functionalities into seven layers. A speech coder resides in the application layer, which is the seventh and highest layer. Imperfections in the transmission are removed in both the physical layer (the first layer) and the transport layer (the fourth layer). The physical layer removes *soft* information, which consists of a probability for the allowed symbols, and renders a sequence of bits to the higher layers. Error control normally resides in the transport layer. However, the error control of the transport layer, as specified by the transmission control protocol (TCP) [14.10], and particularly the automatic repeat requests that TCP uses is generally not appropriate for real-time communication of audiovisual data because of delay. TCP is also rarely used for broadcast and multicast applications to reduce the load on the transmitter. Instead, the user datagram protocol (UDP) [14.11] is used, which means that the coded signal is handed up to the higher network layers without error correction. It is possible that in future systems cross-layer interactions will allow the application layer to receive information about the soft information available at the physical layer.

Handing the received coded signal with its defects directly to the application layer allows the usage of both the inherent redundancy in the signal and our knowledge of the perception of distortion by the user. This leads to coding systems that exhibit a graceful degrada-

tion with increasing error rate. We refer to the chapter on voice over internet protocol (IP) for more detail on techniques that lead to robustness against bit errors and packet loss.

14.2.4 Delay

From coding theory [14.12], we know that optimal coding performance generally requires a delay in the transfer of the message. Long delays are impractical because they are generally associated with methods with high computational and storage requirements, and because in real-time environments (common for speech) the user does not tolerate a long delay.

Significant delay directly affects the quality of a conversation. Impairment to conversations is measurable at one-way delays as low as 100 ms [14.13], although 200 ms is often considered a useful bound.

Echo is perceivable at delays down to 20 ms [14.14]. Imperfections in the network often lead to so-called network echo. Low-delay codecs have been designed to keep the effect of such echo to a minimum, e.g., [14.15]. However, echo cancelation has become commonplace in communication networks. Moreover, packet networks have an inherent delay that requires echo cancelation even for low-delay speech codecs. Thus, for most applications codecs can be designed without consideration of echo.

In certain applications the user may hear both an acoustic signal and a signal transmitted by a network. Examples are flight control rooms and wireless systems for hearing-impaired persons. In this class of applications, coding delays of less than 10 ms are needed to attain an acceptable overall delay.

14.2.5 Computational and Memory Requirements

Economic cost is generally a function of the computational and memory requirements of the coding system. A common measure of computational complexity used in applications is the number of instructions required on a particular silicon device. This is often translated into the number of channels that can be implemented on a single device.

A complicating factor is that speech codecs are commonly implemented on fixed-point signal processing devices. Implementation on a fixed-point device generally takes significant development effort beyond that of the development of the floating-point algorithm.

It is well known that vector quantization facilitates an optimal rate versus quality trade-off. Basic vector quantization techniques require very high computational effort and the introduction of vector quantization in speech coding resulted in promising but impractical codecs [14.16]. Accordingly, significant effort was spent to develop vector quantization structures that facilitate low computational complexity [14.17–19]. The continuous improvement in vector quantization methods and an improved understanding of the advantages of vector quantization over scalar quantization [14.20, 21] has meant that the computational effort of speech codecs has not changed significantly over the past two decades, despite significant improvement in codec performance. More effective usage of scalar quantization and the development of effective lattice vector quantization techniques make it unlikely that the computational complexity of speech codecs will increase significantly in the future.

14.3 A Universal Coder for Speech

In this section, we consider the encoding of a speech signal from a fundamental viewpoint. In information-theoretic terminology, speech is our *source* signal. We start with a discussion of the direct encoding of speech segments, without imposing any structure on the coder. This discussion is not meant to lead directly to a practical coding method (the computational effort would not be reasonable), but to provide an insight into the structure of existing coders. We then show how a signal model can be introduced. The signal model facilitates coding at a reasonable computational cost and the resulting coding paradigm is used by most speech codecs.

14.3.1 Speech Segment as Random Vector

Speech coders generally operate on a sequence of subsequent signal segments, which we refer to as *blocks* (also commonly known as *frames*). Blocks consist generally, but not always, of a fixed number of samples. In the present description of a basic coding system, we divide the speech signal into subsequent blocks of equal length and denote the block length in samples by k. We neglect dependencies across block boundaries, which is not always justified in a practical implementation, but simplifies the discussion; it is generally straightforward to cor-

rect this omission on implementation. We assume that the blocks can be described by k-dimensional random vectors $\boldsymbol{X}^k$ with a probability density function $p_{\boldsymbol{X}^k}(\boldsymbol{x}^k)$ for any $\boldsymbol{x}^k \in \mathbb{R}^k$, the k-dimensional Euclidian space (following convention, we denote random variables by capital letters and realizations by lower case letters).

For the first part of our discussion (Sect. 14.3.2), it is sufficient to assume the existence of the probability density function. It is natural, however, to consider some structure of the probability density $p_{\boldsymbol{X}^k}(\cdot)$ based on the properties of speech. We commonly describe speech in terms of a particular set of sounds (a distinct set of phones). A speech vector then corresponds to one sound from a countable set of speech sounds. We impose the notion that speech consists of a set of sounds on our probabilistic speech description. We can think of each sound as having a particular probability density. A particular speech vector then has one of a set of possible probability densities. Each member probability density of the set has an a *prior* probability, denoted as $p_I(i)$, where i indexes the set. The prior probability $p_I(i)$ is the probability that a random vector $\boldsymbol{X}^k$ is drawn from the particular member probability density i. The overall probability function of the random speech vector $p_{\boldsymbol{X}^k}(\cdot)$ is then a *mixture* of probability density functions

$$p_{\boldsymbol{X}^k}(\boldsymbol{x}^k) = \sum_{i \in \mathcal{A}} p_I(i) p_{\boldsymbol{X}^k|I}(\boldsymbol{x}^k|i) \,, \tag{14.1}$$

where $\mathcal{A}$ is the set of indexes for component densities and $p_{\boldsymbol{X}^k|I}(\cdot|\cdot)$ is the density of component i. These densities are commonly referred to as *mixture components*. If the set of mixture components is characterized by continuous parameters, then the summation must be replaced by an integral.

A common motivation for the mixture formulation of (14.1) is that a good approximation to the true probability density function can be achieved with a mixture of a finite set of probability densities from a particular family. This eliminates the need for the physical motivation. The family is usually derived from a single *kernel* function, such as a Gaussian. The kernel is selected for mathematical tractability.

If a mixture component does correspond to a physically reasonable speech sound, then it can be considered a statistical *model* of the signal. As described in Sect. 14.3.4, it is possible to interpret existing speech coding paradigms from this viewpoint. For example, linear prediction identifies a particular autoregressive model appropriate for a block. Each of the autoregressive models of speech has a certain prior probability and this in turn leads to an overall probability for the speech vector. According to this interpretation, mixture models have long been standard tools in speech coding, even if this was not explicitly stated.

The present formalism does not impose stationarity conditions on the signal within the block. In the mixture density, it is reasonable to include densities that correspond to signal transitions. In practice, this is not common, and the probability density functions are usually defined based on the definition that the signal is stationary within a block. On the other hand, the assumption that all speech blocks are drawn from the same distribution is implicit in the commonly used coding methods. It is consistent with our neglect of interblock dependencies. Thus, if we consider the speech signal to be a vector signal, then we assume stationarity for this vector signal (which is a rather inaccurate approximation). Strictly speaking, we do not assume ergodicity, as averaging over a database is best interpreted an averaging over an ensemble of signals, rather than time averaging over a single signal.

14.3.2 Encoding Random Speech Vectors

To encode observed speech vectors $\boldsymbol{x}^k$ that form *realizations* of the random vector $\boldsymbol{X}^k$, we use a speech codebook $C_{\boldsymbol{X}^k}$ that consists of a countable set of k-dimensional vectors (the code vectors). We can write $C_{\boldsymbol{X}^k} = \{\boldsymbol{c}_q^k\}_{q \in \mathcal{Q}}$, where $\boldsymbol{c}_q^k \in \mathbb{R}^k$ and $\mathcal{Q}$ is a countable (but not necessarily finite) set of indices. A decoded vector is simply the entry of the codebook that is pointed to by a transmitted index.

The encoding with codebook vectors results in the removal of both redundancy and irrelevancy. It removes irrelevancy by introducing a reduced precision version of the vector $\boldsymbol{x}^k$, i. e., by quantizing $\boldsymbol{x}^k$. The quantized vector requires fewer bits to encode than the unquantized vector. The mechanism of the redundancy removal depends on the coding method and will be discussed in Sect. 14.3.3.

We consider the speech vector, $\boldsymbol{X}^k$, to have a continuous probability density function in $\mathbb{R}^k$. Thus, coding based on the finite-size speech codebook $C_{\boldsymbol{X}^k}$ introduces *distortion*. To minimize the distortion associated with the coding, the encoder selects the code vector (codebook entry) $\boldsymbol{c}_q^k$ that is nearest to the observed vector $\boldsymbol{x}^k$ according to a particular distortion measure,

$$q = \operatorname{argmin}_{q' \in \mathcal{Q}} d\left(\boldsymbol{x}^k, \boldsymbol{c}_{q'}^k\right) . \tag{14.2}$$

Quantization is the operation of finding the nearest neighbor in the codebook. The set of speech vectors

that is mapped to a particular code vector $\boldsymbol{c}_q^k$ is called a *quantization cell* or *Voronoi region*. We denote the Voronoi region as $\mathcal{V}_q$,

$$\mathcal{V}_q = \left\{\boldsymbol{x}^k : d\left(\boldsymbol{x}^k, \boldsymbol{c}_q^k\right) < d\left(\boldsymbol{x}^k, \boldsymbol{c}_m^k\right) \forall_{m \neq q}\right\} , \qquad (14.3)$$

where we have ignored that generally points exist for which the inequality is not strict. These are boundary points that can be assigned to any of the cells that share the boundary.

Naturally, the average [averaged over $p_{\boldsymbol{X}^k}(\cdot)$] distortion of the decoded speech vectors differs for different codebooks. A method for designing a coder is to find the codebook, i. e., the set $C_{\boldsymbol{X}^k} = \{\boldsymbol{c}_q^k\}_{q \in \mathcal{Q}}$, that minimizes the average distortion over the speech probability density, given a constraint on the transmission rate. It is not known how to solve this problem in a general manner. Iterative methods (the Lloyd algorithm and its variants, e.g., [14.22–24]) have been developed for the case where $|\mathcal{Q}|$ (the *cardinality* or number of vectors in $C_{\boldsymbol{X}^k}$) is finite. The iterative approach is not appropriate for our present discussion for two reasons. First, we ultimately are interested in structured quantizers that allow us to approximate the optimal codebook and structure is difficult to determine from the iterative method. Second, as we will see below for the constrained-entropy case, practical codebooks do not necessarily have finite cardinality. Instead of the iterative approach, we use an approach where we make simplifying assumptions, which are asymptotically accurate for high coding rates.

14.3.3 A Model of Quantization

To analyze the behavior of the speech codebook, we construct a model of the quantization (encoding–decoding) operation. (This *quantization* model is not to be confused with the probabilistic *signal* model described in the next subsection.) Thus, we make the quantization problem mathematically tractable. For simplicity, we use the squared error criterion (Sect. 14.5 shows that this criterion can be used over a wide range of coding scenarios). We also make the standard assumption that the quantization cells are convex (for any two points in a cell, all points on the line segment connecting the two points are in the cell). To construct our encoding–decoding model, we make three additional assumptions that cannot always be justified:

1. The density $p_{\boldsymbol{X}^k}(\boldsymbol{x}^k)$ is constant within each quantization cell. This implies that the probability that a speech vector is inside a cell with index q is

$$p_Q(q) = V_q p_{\boldsymbol{X}^k}(\boldsymbol{x}^k), \boldsymbol{x}^k \in \mathcal{V}_q, \qquad (14.4)$$

where V_q is the volume of the k-dimensional cell.

2. The average distortion for speech data falling within cell q is

$$D_q = C V_q^{\frac{2}{k}} , \qquad (14.5)$$

where C is a constant. The assumption made in (14.5) essentially means that the cell shape is fixed. *Gersho* [14.25], conjectured that this assumption is correct for optimal codebooks.

3. We assume that the countable set of code vectors $C_{\boldsymbol{X}^k}$ can be represented by a code-vector density, denoted as $g(\boldsymbol{x}^k)$. This means that the cell volume now becomes a function of $\boldsymbol{x}^k$ rather than the cell index q; we replace V_q by $V(\boldsymbol{x}^k)$. To be consistent we must equate the density with the inverse of the cell volume:

$$g(\boldsymbol{x}^k) = \frac{1}{V(\boldsymbol{x}^k)} . \qquad (14.6)$$

The third assumption also implies that we can replace D_q by $D(\boldsymbol{x}^k)$.

The three assumptions listed above lead to solutions that can generally be shown to hold asymptotically in the limit of infinite rate. The theory has been observed to make reasonable predictions of performance for practical quantizers at rates down to two bits per dimension [14.26, 27], but we do not claim accuracy here. The theory serves as a vehicle to understand quantizer behavior and *not* as an accurate predictor of performance.

The code-vector density $g(\boldsymbol{x}^k)$ of our quantization model replaces the set of code vectors as the description of the codebook. Our objective of finding the codebook that minimizes the average distortion subject to a rate constraint has become the objective of finding the optimal density $g(\boldsymbol{x}^k)$ that minimizes the distortion

$$\begin{aligned} D &= \int D(\boldsymbol{x}^k) p_{\boldsymbol{X}^k}(\boldsymbol{x}^k) \mathrm{d}\boldsymbol{x}^k \\ &= C \int V(\boldsymbol{x}^k)^{\frac{2}{k}} p_{\boldsymbol{X}^k}(\boldsymbol{x}^k) \mathrm{d}\boldsymbol{x}^k \\ &= C \int g(\boldsymbol{x}^k)^{-\frac{2}{k}} p_{\boldsymbol{X}^k}(\boldsymbol{x}^k) \mathrm{d}\boldsymbol{x}^k , \end{aligned} \qquad (14.7)$$

subject to a rate constraint.

Armed with our quantization model, we now attempt to find the optimal density $g(\boldsymbol{x}^k)$ (the optimal codebook) for encoding speech. We consider separately two commonly used constraints on the rate: a given fixed rate and a given average rate. As mentioned in Sect. 14.2.1, the former rate constraint applies to circuit-switched networks and the latter rate constraint represents situations

where the rate can be varied continuously, such as, for example, in storage applications and packet networks.

We start with the fixed-rate requirement, where each codebook vector $\boldsymbol{c}_q^k$ is encoded with a codeword of a fixed number of bits. This is called *constrained-resolution* coding. If we use a rate of R bits per speech vector then we have a codebook cardinality of $N = 2^R$ and the density $g(\boldsymbol{x}^k)$ must be consistent with this cardinality:

$$N = \int_{\mathbb{R}^k} g(\boldsymbol{x}^k)\,\mathrm{d}\boldsymbol{x}^k \,. \tag{14.8}$$

We have to minimize the average distortion of (14.7) subject to the constraint (14.8) (i. e., subject to given N). This constrained optimization problem is readily solved with the calculus of variations. The solution is

$$\begin{aligned} g(\boldsymbol{x}^k) &= N \frac{p_{\boldsymbol{X}^k}(\boldsymbol{x}^k)^{\frac{k}{k+2}}}{\int p_{\boldsymbol{X}^k}(\boldsymbol{x}^k)^{\frac{k}{k+2}}\,\mathrm{d}\boldsymbol{x}^k} \\ &= 2^R \frac{p_{\boldsymbol{X}^k}(\boldsymbol{x}^k)^{\frac{k}{k+2}}}{\int p_{\boldsymbol{X}^k}(\boldsymbol{x}^k)^{\frac{k}{k+2}}\,\mathrm{d}\boldsymbol{x}^k} \\ &= 2^R \underline{p_{\boldsymbol{X}^k}(\boldsymbol{x}^k)^{\frac{k}{k+2}}} \,, \end{aligned} \tag{14.9}$$

where the underlining denotes normalization to unit integral over $\mathbb{R}^k$ and where R is the bit rate per speech vector. Thus, our encoding–decoding model suggests that, for constrained-resolution coding, the density of the code vectors varies with the data density. At dimensionalities $k >> 1$ the density of the code vectors approximates a simple scaling of the probability density of the speech vectors since $k/(k+2) \to 1$ with increasing k.

In the constrained-resolution case, *redundancy* is removed by placing the codebook vectors such that they reflect the density of the data vectors. For example, as shown by (14.9), regions of $\mathbb{R}^k$ without data have no vectors placed in them. This means no codewords are used for regions that have no data. If we had placed codebook vectors there, these would have been redundant. Note that scalar quantization of the k-dimensional random vector $\boldsymbol{X}^k$ would do precisely that. Similarly, regions of low data density get relatively few code vectors, reducing the number of codewords spent in such regions.

Next, we apply our quantization model to the case where the average rate is constrained. That is, the codeword length used to encode the cell indices q varies. Let us denote the random index associated with the random vector $\boldsymbol{X}^k$ as Q. The source coding theorem [14.12] tells us the lowest possible average rate for uniquely (so it can be decoded) encoding the indices with separate codewords is within one bit of the index entropy (in bits)

$$H(Q) = -\sum_{q\in\mathcal{Q}} p_Q(q)\log_2[p_Q(q)] \,. \tag{14.10}$$

The entropy can be interpreted as the average of a bit allocation, $-\log_2[p_Q(q)]$, for each index q. Neglecting the aforementioned *within one bit*, the average rate constraint is $H(Q) = R$, where R is the selected rate. For this reason, this coding method is known as *constrained-entropy* coding. This neglect is reasonable as the difference can be made arbitrarily small by encoding sequences of indices, as in arithmetic coding [14.28], rather than single indices. We minimize the distortion of (14.7) subject to the constraint (14.10), i. e., subject to given $H(Q) = R$. Again, the constrained optimization problem is readily solved with the calculus of variations. In this case the solution is

$$g(\boldsymbol{x}^k) = 2^{H(Q)-h(\boldsymbol{X}^k)} = 2^{R-h(\boldsymbol{X}^k)} \,, \tag{14.11}$$

where $h(\boldsymbol{X}^k) = -\int p_{\boldsymbol{X}^k}(\boldsymbol{x}^k)\log_2[p_{\boldsymbol{X}^k}(\boldsymbol{x}^k)]\,\mathrm{d}\boldsymbol{x}^k$ is the *differential entropy* of $\boldsymbol{X}^k$ in bits, and where $H(Q)$ is specified in bits. It is important to realize that special care must be taken if $p_{\boldsymbol{X}^k}(\cdot)$ is singular, i. e., if the data lie on a manifold.

Equation (14.11) implies that the the code vector density is uniform across $\mathbb{R}^k$. The number of code vectors is countably infinite despite the fact that the rate itself is finite. The codeword length $-\log_2[p_Q(q)]$ increases very slowly with decreasing probability $p_Q(q)$ and, roughly speaking, long codewords make no contribution to the mean rate.

In the constrained-entropy case, redundancy is removed through the lossless encoding of the indices. Given the probabilities of the code vectors, (ideal) lossless coding provides the most efficient bit assignment that allows unique decoding, and this rate is precisely the entropy of the indices. Code vectors in regions of high probability density receive short codewords and code vectors in regions of low probability density receive long codewords.

An important result that we have found for both the constrained-resolution and constrained-entropy cases is that the structure of the codebook is independent of the overall rate. The code-vector density simply increases as 2^R (cf. (14.9) and (14.11), respectively) anywhere in $\mathbb{R}^k$. Furthermore, for the constrained-entropy case, the code vector density depends only through the global variable $h(\boldsymbol{X}^k)$ on the probability density.

14.3.4 Coding Speech with a Model Family

Although the quantization model of Sect. 14.3.3 provides interesting results, a general implementation of

a codebook for the random speech vector $\boldsymbol{X}^k$ leads to practical problems, except for small k. For the constrained-resolution case, larger values of k lead to codebook sizes that do not allow for practical training procedures for storage on conventional media. For the constrained-entropy case, the codebook itself need not be stored, but we need access to the probability density of the codebook entries to determine the corresponding codewords (either offline or through computation during encoding). We can resolve these practical coding problems by using a *model* of the density. Importantly, to simplify the computational effort, we do *not* assume that the model is an accurate representation of the density of the speech signal vector, we simply make a best effort given the tools we have.

The model-based approach towards reducing computational complexity is suggested by the mixture model that we discussed in Sect. 14.3.1. If we classify each speech vector first as corresponding to a particular sound, then we can specify a probability density for that sound. A signal model specifies the probability density, typically by means of a formula for the probability density. The probability densities of the models are typically selected to be relatively simple. The signal models reduce computational complexity, either because they reduce codebook size or because the structural simplicity of the model simplifies the lossless coder. We consider models of a similar structure to be member of a *model family*. The selection of a particular model from the family is made by specifying *model parameters*.

Statistical signal models are commonly used in speech coding, with autoregressive modeling (generally referred to as linear prediction coding methods) perhaps being the most common. In this section, we discuss the selection of a particular model from the model family (i.e., the selection of the model parameters) and the balance in bit allocation between the model and the specification of the speech vector.

Our starting point is that a family of signal models is available for the coding operation. The model family can be any model family that provides a probability assignment for the speech vector $\boldsymbol{x}^k$. We discuss relevant properties for coding with signal models. We do not make the assumption that the resulting coding method is close to a theoretical performance bound on the rate versus distortion trade-off. As seid, we also do not make an assumption about the appropriateness of the signal model family for the speech signal. The model probabilities may not be accurate. However it is likely that models that are based on knowledge of speech production result in better performance.

The reasoning below is based on the early descriptions of the minimum description length (MDL) principle for finding signal models [14.29–31]. These methods separate a code for the model and a code for the signal realization, making them relevant to practical speech coding methods whereas later MDL methods use a single code. Differences from the MDL work include a stronger focus on distortion, and the consideration of the constrained-resolution case, which is of no interest to modeling theory.

Constrained-Entropy Case

First we consider the constrained-entropy case, i. e., we consider the case of a uniform codebook. Each speech vector is encoded with a codebook where each cell is of identical volume, which we denote as V. Let the model distribution be specified by a set of model parameters, θ. We consider the models to have a probability density, which means that a particular parameter set θ corresponds to a particular realization of a random parameter vector Θ. We write the probability density of $\boldsymbol{X}^k$ assuming the particular parameter set θ as $p_{\boldsymbol{X}^k;\Theta}(\boldsymbol{x}^k|\theta)$. The corresponding overall model density is

$$\tilde{p}_{\boldsymbol{X}^k}(\boldsymbol{x}^k) = -\sum_{\theta} p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta) \cdot p_{\Theta}(\theta)\,, \qquad (14.12)$$

where the summation is over all parameter sets. The advantage of selecting and then using models $p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta)$ from the family over using the composite model density $\tilde{p}_{\boldsymbol{X}^k}(\boldsymbol{x}^k)$ is a decrease of the computational effort.

The quantization model of Sect. 14.3.3 and in particular (14.4) and (14.10), show that the constrained-entropy encoding of a vector $\boldsymbol{x}^k$ assuming the model with parameters θ requires $-\log_2[Vp_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta)]$ bits. In addition, the decoder must receive side information specifying the model.

Let $\hat{\theta}(\boldsymbol{x}^k)$ be the parameter vector that maximizes $p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta)$ and, thus, minimizes the bit allocation $-\log_2[Vp_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta)]$. That is, $p_{\boldsymbol{X}^k|\Theta}[\boldsymbol{x}^k|\hat{\theta}(\boldsymbol{x}^k)]$ is the maximum-likelihood model (from the family) for encoding the speech vector $\boldsymbol{x}^k$. The random speech vectors $\boldsymbol{X}^k$ do not form a countable set and as a result the random parameter vector $\hat{\Theta}(\boldsymbol{X}^k)$ generally does not form a countable set for conventional model families such as autoregressive models. To encode the model, we must discretize it.

To facilitate transmission of the random model index, J, the model parameters must be quantized and we write the random parameter set corresponding to random index J as $\theta(J)$. If $p_J(j)$ is a prior probability of the model index, the overall bit allocation for the vector

$\boldsymbol{x}^k$ when encoded with model j is

$$\begin{aligned} l &= -\log_2[p_J(j)] - \log_2\left\{V(\boldsymbol{x}^k)\, p_{\boldsymbol{X}^k|\Theta}[\boldsymbol{x}^k|\theta(j)]\right\} \\ &= -\log_2[p_J(j)] + \log_2\left(\frac{p_{\boldsymbol{X}^k|\Theta}[\boldsymbol{x}^k|\hat{\theta}(\boldsymbol{x}^k)]}{p_{\boldsymbol{X}^k|\Theta}[\boldsymbol{x}^k|\theta(j)]}\right) \\ &\quad - \log_2[V(\boldsymbol{x}^k)\, p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\hat{\theta})]\,, \end{aligned} \tag{14.13}$$

where the term $\log_2\left(\frac{p_{\boldsymbol{X}^k|\Theta}[\boldsymbol{x}^k|\hat{\theta}(\boldsymbol{x}^k)]}{p_{\boldsymbol{X}^k|\Theta}[\boldsymbol{x}^k|\theta(j)]}\right)$ represents the additional (excess) bit allocation required to encode $\boldsymbol{x}^k$ with model j over the bit allocation required to encode $\boldsymbol{x}^k$ with the true maximum-likelihood model from the model family.

With some abuse of notation, we denote by $j(\boldsymbol{x}^k)$ the function that provides the index for a given speech vector $\boldsymbol{x}^k$. In the following, we assume that the functions $\theta(j)$ and $j(\boldsymbol{x}^k)$ minimize l. That is, we quantize θ so as to minimize the total number of bits required to encode $\boldsymbol{x}^k$.

We are interested in the bit allocation that results from averaging over the probability density $p_{\boldsymbol{X}^k}(\cdot)$ of the speech vectors,

$$\begin{aligned} \mathrm{E}\{\mathsf{L}\} = &-\mathrm{E}\left\{\log_2[p_J(j(\boldsymbol{X}^k))]\right\} \\ &-\mathrm{E}\left\{\log_2\left(\frac{p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{X}^k|\theta(j(\boldsymbol{X}^k)))}{p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{X}^k|\hat{\theta}(\boldsymbol{X}^k))}\right)\right\} \\ &-\mathrm{E}\left\{\log_2\left[V(\boldsymbol{X}^k)\, p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{X}^k|\hat{\theta}(\boldsymbol{X}^k))\right]\right\}, \end{aligned} \tag{14.14}$$

where $\mathrm{E}\{\cdot\}$ indicates averaging over the speech vector probability density and where L is the random bit allocation that has I as realization. In (14.14), the first term describes the mean bit allocation to specify the model, the second term specifies the mean excess in bits required to encode $\boldsymbol{X}^k$ assuming $\theta(j(\boldsymbol{x}^k))$ instead of assuming the optimal $\hat{\theta}(\boldsymbol{x}^k)$, and the third term specifies the mean number of bits required to encode $\boldsymbol{X}^k$ if the optimal model is available. Importantly, only the third term contains the cell volume that determines the mean distortion of the speech vectors through (14.7).

Assuming validity of the encoding model of Sect. 14.3.3, the optimal trade-off between the bit allocation for the model index and the bit allocation for the speech vectors $\boldsymbol{X}^k$ depends only on the mean of

$$\begin{aligned} \eta = &-\log_2\{p_J[j(\boldsymbol{x}^k)]\} \\ &-\log_2\left(\frac{p_{\boldsymbol{X}^k|\Theta}[\boldsymbol{x}^k|\theta(j(\boldsymbol{x}^k))]}{p_{\boldsymbol{X}^k|\Theta}[\boldsymbol{x}^k|\hat{\theta}(\boldsymbol{x}^k)]}\right), \end{aligned} \tag{14.15}$$

which is referred to as the [14.32]. The goal is to find the functions $\theta(\cdot)$ and $j(\cdot)$ that minimize the index of resolvability over the ensemble of speech vectors. An important consequence of our logic is that these functions, and therefore the rate allocation for the model index, are dependent only on the excess rate and the probability of the quantized model. As the third term of (14.14) is missing, no relation to the speech distortion exists. That is *the rate allocation for the model index J is independent of distortion and overall bit rate*. While the theory is based on assumptions that are accurate only for high bit rates, this suggests that the bit allocation for the parameters becomes proportionally more important at low rates.

The fixed entropy for the model index indicates, for example, that for the commonly used linear-prediction-based speech coders, the rate allocation for the linear prediction parameters is independent of the overall rate of the coder. As constrained-entropy coding is not commonly used for predictive coding, this result is not immediately applicable to conventional speech coders. However, the new result we derive below is applicable to such coders.

Constrained-Resolution Case

Most current speech coders were designed with a constrained-resolution (fixed-rate) constraint, making it useful to study modeling in this context. We need some preliminary results. For a given model, with parameter set θ, and optimal code vector density, the average distortion over a quantization cell centered at location $\boldsymbol{x}^k$ can be written

$$\begin{aligned} D(\boldsymbol{x}^k) &= CV(\boldsymbol{x}^k)^{\frac{2}{k}} \\ &= Cg(\boldsymbol{x}^k)^{-\frac{2}{k}} \\ &= CN^{-\frac{2}{k}}\left[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta)^{\frac{k}{k+2}}\right]^{-\frac{2}{k}}, \end{aligned} \tag{14.16}$$

where we have used (14.7) and (14.9). We take the expectation of (14.16) with respect to the true probability density function $p_{\boldsymbol{X}^k}(\boldsymbol{x}^k)$ and obtain the mean distortion for the constrained-resolution case:

$$D_{\mathrm{CR}} = CN^{-\frac{2}{k}}\mathrm{E}\left\{\left[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{X}^k|\theta)^{\frac{k}{k+2}}\right]^{-\frac{2}{k}}\right\}. \tag{14.17}$$

Equation (14.17) can be rewritten as

$$\begin{aligned} \frac{2}{k}\log_2(N) = &\log_2\left(\mathrm{E}\left\{\left[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{X}^k|\theta)^{\frac{k}{k+2}}\right]^{-\frac{2}{k}}\right\}\right) \\ &-\log_2\left(\frac{D_{\mathrm{CR}}}{C}\right). \end{aligned} \tag{14.18}$$

We assume that k is sufficiently large that, in the region where $p_{\boldsymbol{X}^k}(\boldsymbol{x}^k)$ is significant, we can use the expansion

$u \approx 1 + \log(u)$ for the term $[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta)^{k/(k+2)}]^{-2/k}$ and write

$$\begin{aligned}
&\log\left(\mathrm{E}\left\{\left[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{X}^k|\theta)^{\frac{k}{k+2}}\right]^{-\frac{2}{k}}\right\}\right)\\
&\approx \log\left(1-\frac{2}{k}\mathrm{E}\left\{\log\left[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{X}^k|\theta)^{\frac{k}{k+2}}\right]\right\}\right)\\
&\approx -\frac{2}{k}\mathrm{E}\left\{\log\left[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{X}^k|\theta)^{\frac{k}{k+2}}\right]\right\} . \qquad (14.19)
\end{aligned}$$

Having completed the preliminaries, we now consider the encoding of a speech vector $\boldsymbol{x}^k$. Let $L_{(m)}$ be the fixed bit allocation for the model index. The total rate is then

$$\begin{aligned}
L &= L_{(m)} + L(\boldsymbol{x}^k)\\
&= L_{(m)} + \log_2(N)\\
&= L_{(m)} - \mathrm{E}\left\{\log_2\left[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{X}^k|\theta)^{\frac{k}{k+2}}\right]\right\}\\
&\quad -\frac{k}{2}\log_2\left(\frac{D_{\mathrm{CR}}}{C}\right) . \qquad (14.20)
\end{aligned}$$

The form of (14.20) shows that, given the assumptions made, we can define an *equivalent codeword length* $\log_2[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{X}^k|\theta)^{k/(k+2)}]$ for each speech codebook entry. The equivalent codeword length represents the spatial variation of the distortion. Note that this equivalent codeword length does *not* correspond to the true codeword length of the speech vector codebook, which is fixed for the constrained-resolution case. For a particular codebook vector $\boldsymbol{x}^k$, the equivalent codeword length is

$$\begin{aligned}
L &= L_{(m)} - \log_2\left[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta)^{\frac{k}{k+2}}\right]\\
&\quad -\frac{k}{2}\log_2\left(\frac{D_{\mathrm{CR}}}{C}\right) . \qquad (14.21)
\end{aligned}$$

Similarly to the constrained-entropy case, we can decompose (14.21) into a rate component that relates to the encoding of the model parameters, a component that describes the excess equivalent rate resulting from limiting the precision of the model parameters, and a rate component that relates to optimal encoding with optimal (uncoded) model parameters:

$$\begin{aligned}
L &= L_{(m)} - \log_2(p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta(j))^{\frac{k}{k+2}}) - \frac{k}{2}\log_2\left(\frac{D_{\mathrm{CR}}}{C}\right)\\
&= L_{(m)} - \log_2\left(\frac{p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta(j))^{\frac{k}{k+2}}}{p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\hat{\theta}(\boldsymbol{x}^k))^{\frac{k}{k+2}}}\right)\\
&\quad -\log_2\left[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\hat{\theta}(\boldsymbol{x}^k))^{\frac{k}{k+2}}\right] - \frac{k}{2}\log_2\left(\frac{D_{\mathrm{CR}}}{C}\right) ,
\end{aligned}$$

(14.22)

We can identify the last two terms as the bit allocation for $\boldsymbol{x}^k$ for the optimal constrained-resolution model for the speech vector $\boldsymbol{x}^k$. The second term is the excess equivalent bit allocation required to encode the speech vector with model j over the bit allocation required for the optimal model from the model family. The first two terms determine the trade-off between the bits spent on the model, and the bits spent on the speech vectors. These two terms form the index of resolvability for the constrained-resolution case:

$$\eta = L_{(m)} - \log_2\left(\frac{p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta(j))^{\frac{k}{k+2}}}{p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\hat{\theta}(\boldsymbol{x}^k))^{\frac{k}{k+2}}}\right) . \qquad (14.23)$$

As for the constrained-entropy case, the optimal set of functions $\theta(j)$ and $j(\boldsymbol{x}^k)$ (and, therefore, the bit allocation for the model) are dependent only on the speech vector density for the constrained-resolution case. The rate for the model is independent of the distortion selected for the speech vector and of the overall rate. With increasing k, the second term in (14.23) and (14.15) becomes identical. That is the expression for the excess rate for using the quantized model parameters corresponding to model j instead of the optimal parameters is identical.

The independence of the model-parameter bit allocation of the overall codec rate for the constrained-entropy case is of great significance for practical coding systems. We emphasize again that this result is valid only under the assumptions made in Sect. 14.3.3. We expect the independence to break down at lower rates, where the codebook $C_{\boldsymbol{X}^k}$ describing the speech cannot be approximated by a density.

Table 14.1 Bit rates of the AMR-WB coder [14.4]

Rate (bits)	6.6	8.85	12.65	14.25	15.85	18.25	19.85	23.05
AR model	36	46	46	46	46	46	46	46
Pitch parameter	23	26	30	30	30	30	30	30
Excitation	48	80	144	176	208	256	288	352

The results described in this section are indeed supported, at least qualitatively, by the configuration of practical coders. Table 14.1 shows the most important bit allocations used in the adaptive-multirate wideband (AMR-WB) speech coder [14.4]. The AMR-WB coder is a constrained-resolution coder. It is seen that the design of the codec satisfies the predicted behavior: the bit allocation for the model parameters is essentially independent of the rate of the codec, except at low rates.

Model-Based Coding

In signal-model-based coding we assume the family is known to the encoder and decoder. An index to the specific model is transmitted. Each model corresponds to a unique speech-domain codebook. The advantage of the model-based approach is that the structure of the density is simplified (which is advantageous for constrained-entropy coding) and that the required number of codebook entries for the constrained-resolution case is smaller. This facilitates searching through the codebook and/or the definition of the lossless coder.

The main result of this section is that we can determine the set of codebooks for the models independently of the overall rate (and speech-vector distortion). The result is consistent with existing results. The result of this section leads to fast codec design as there is no need to check the best trade-off in bit allocation between model and signal quantization.

When encoding with a model-based coding it is advantageous first to identify the *best* model, encode the model index j, and then encode the signal using codebook $C_{\boldsymbol{X}^k,j}$ that is associated with that particular model j. The model selection can be made based on the index of resolvability.

14.4 Coding with Autoregressive Models

We now apply the methods of Sect. 14.3 to a practical model family. Autoregressive model families are commonly used in speech coding. In speech coding this class of coders is generally referred to as being based on *linear prediction*. We discuss coding based on a family that consists of a set of autoregressive models of a particular order (denoted as p). To match current practice, we consider the constrained-resolution case.

We first formulate the index of resolvability in terms of a spectral formulation of the autoregressive model. We show that this corresponds to the definition of a distortion measure for the model parameters. The distortion measure is approximated by the commonly used Itakura–Saito and log spectral distortion measures. Thus, starting from a squared error criterion for the speech signal, we obtain the commonly used (e.g., [14.33–37]) distortion measures for the linear-prediction parameters. Finally, we show that our reasoning leads to an estimate for the bit allocation for the model. We discuss how this result relates to results on autoregressive model estimation.

14.4.1 Spectral-Domain Index of Resolvability

Our objective is to encode a particular speech vector $\boldsymbol{x}^k$ using the autoregressive model. To facilitate insight, it is beneficial to make a spectral formulation of the problem. To this purpose, we assume that k is sufficiently large to neglect edge effects. Thus, we neglect the difference between circular and linear convolution.

The autoregressive model assumption implies that $\boldsymbol{x}^k$ has a multivariate Gaussian probability density

$$p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta) = \frac{1}{\sqrt{2\pi\det(\mathbf{R}_\theta)}}\exp\left(-\frac{1}{2}\boldsymbol{x}^{kT}\mathbf{R}_\theta^{-1}\boldsymbol{x}^k\right) . \quad (14.24)$$

$\mathbf{R}_\theta$ is the *model* autocorrelation matrix

$$\mathbf{R}_\theta = \mathbf{A}^{-1}\mathbf{A}^{-\mathrm{H}} , \quad (14.25)$$

where $\mathbf{A}$ a lower-triangular Toeplitz matrix with first column $\sigma[1, a_1, a_2, \cdots, a_p, 0, \cdots, 0]^{\mathrm{T}}$, where the a_i are the autoregressive model parameters (linear-prediction parameters), and p is the autoregressive model order and the superscript H is the Hermitian transpose. Thus, the set of model parameters is $\theta = \{\sigma, a_1, \cdots, a_p\}_{i=1,\cdots,p}$. We note that typically $p = 10$ for 8 kHz sampling rate and $p = 16$ for 12 kHz and 16 kHz sampling rate.

When k is sufficiently large, we can perform our analysis in terms of power spectral densities. The transfer function of the autoregressive model is

$$A(z)^{-1} = \frac{\sigma}{1 + a_1 z^{-1} + \cdots + a_p z^{-p}} , \quad (14.26)$$

where σ is a gain. This corresponds to the model power spectral density

$$\mathbf{R}_\theta(z) = |A(z)|^{-2} \,. \tag{14.27}$$

In the following, we make the standard assumption that $A(z)$ is minimum-phase.

Next we approximate (14.24) in terms of power spectral densities and the transfer function of the autoregressive model. Using *Szegö*'s theorem [14.38], it is easy to show that, asymptotically in k,

$$\det(\mathbf{R}_\theta) = \exp\left\{\frac{k}{2\pi}\int_0^{2\pi}\log\left[\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})\right]\mathrm{d}\omega\right\} \,. \tag{14.28}$$

We also use the asymptotic equality

$$\begin{aligned}\frac{1}{2}\boldsymbol{x}^{kT}\mathbf{R}_\theta^{-1}\boldsymbol{x}^k &= \frac{1}{4\pi}\int_0^{2\pi}\frac{|x(\mathrm{e}^{\mathrm{i}\omega})|^2}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})}\mathrm{d}\omega \\ &= \frac{k}{4\pi}\int_0^{2\pi}\frac{R_x(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})}\mathrm{d}\omega \,, \end{aligned} \tag{14.29}$$

where $x(z) = \sum_{i=0}^{k-1} x_i z^{-i}$ for $\boldsymbol{x}^k = (x_1, \cdots, x_k)$ and $R_x(\mathrm{e}^{\mathrm{i}\omega}) = \frac{1}{k}|x(\mathrm{e}^{\mathrm{i}\omega})|^2$.

Equations (14.28) and (14.29) can be used to rewrite the multivariate density of (14.24) in terms of power spectral densities. It is convenient to write the log density:

$$\begin{aligned}&\log[p_{\boldsymbol{X}^k|\Theta}(\boldsymbol{x}^k|\theta)] \\ &= -\frac{1}{2}\log(2\pi) - \frac{k}{4\pi}\int_0^{2\pi}\log(\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega}))\mathrm{d}\omega \\ &\quad - \frac{k}{4\pi}\int_0^{2\pi}\frac{R_x(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})}\mathrm{d}\omega \,. \end{aligned} \tag{14.30}$$

We use (14.30) to find the index of resolvability for the constrained-resolution case. We make the approximation that k is sufficiently large that it is reasonable to approximate the exponent $k/(k+2)$ by unity in (14.23). This implies that we do not have to consider the normalization in this equation. Inserting (14.30) into (14.23) results in

$$\begin{aligned}\eta = L_{(m)} + \frac{k}{4\pi}\int_0^{2\pi}\Bigg[&-\log\left(\frac{R_{\hat\theta}(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})}\right) \\ &+\frac{R_x(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})} - \frac{R_x(\mathrm{e}^{\mathrm{i}\omega})}{R_{\hat\theta}(\mathrm{e}^{\mathrm{i}\omega})}\Bigg]\mathrm{d}\omega \,. \end{aligned} \tag{14.31}$$

The maximum-likelihood estimate of the autoregressive model $\hat\theta$ given a data vector $\boldsymbol{x}^k$ is a well-understood problem, e.g., [14.39, 40]. The predictor parameter estimate of the standard Yule–Walker solution method has the same asymptotic density as the maximum-likelihood estimate [14.41].

To find the optimal bit allocation for the model we have to minimize the expectation of (14.31) over the ensemble of all speech vectors. We study the behavior of this minimization. For notational convenience we define a cost function

$$\begin{aligned}\psi(\theta,\hat\theta) = \frac{k}{4\pi}\int_0^{2\pi}\Bigg[&-\log\left(\frac{R_{\hat\theta}(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})}\right) \\ &+\frac{R_x(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})} - \frac{R_x(\mathrm{e}^{\mathrm{i}\omega})}{R_{\hat\theta}(\mathrm{e}^{\mathrm{i}\omega})}\Bigg]\mathrm{d}\omega \,. \end{aligned} \tag{14.32}$$

Let θ be a particular model from a countable model set $\mathcal{C}_\Theta(L_{(m)})$ with a bit allocation $L_{(m)}$ for the model. Finding the optimal model set $\mathcal{C}_\Theta(L_{(m)})$ is then equivalent to

$$\begin{aligned}&\min_{L_{(m)}\in\mathbb{N}} \mathrm{E}[\eta] \\ &= \min_{L_{(m)}\in\mathbb{N}}\left\{L_{(m)} + \min_{\mathcal{C}_\Theta(L_{(m)})}\mathrm{E}\Big[\min_{\theta\in\mathcal{C}_\Theta(L_{(m)})}\psi(\theta,\hat\theta)\Big]\right\} \,. \end{aligned} \tag{14.33}$$

If we write

$$D(L_{(m)}) = \min_{\mathcal{C}_\Theta(L_{(m)})}\mathrm{E}\Big[\min_{\theta\in\mathcal{C}_\Theta(L_{(m)})}\psi(\theta,\hat\theta)\Big] \tag{14.34}$$

then (14.33) becomes

$$\min_{L_{(m)}\in\mathbb{N}}\mathrm{E}[\eta] = \min_{L_{(m)}\in\mathbb{N}}\left[L_{(m)} + D(L_{(m)})\right] \,. \tag{14.35}$$

If we interpret $D(L_{(m)})$ as a minimum mean distortion, minimizing (14.35) is equivalent to finding a particular point on a rate-distortion curve. We can minimize the cost function of (14.34) for all $L_{(m)}$ and then select the $L_{(m)}$ that minimizes the overall expression of (14.35). Thus only one particular distortion level, corresponding to one particular rate, is relevant to our speech coding system. This distortion–rate pair for the model is dependent on the distribution of the speech models. Assuming that $D(L_{(m)})$ is once differentiable towards $L_{(m)}$, then (14.35) shows that its derivative should be -1 at the optimal rate for the model.

14.4.2 A Criterion for Model Selection

We started with the notion of using an autoregressive model family to quantize the speech signal. We found

that we could do so by first finding the maximum-likelihood estimate $\hat{\theta}$ of the autoregressive model parameters, then selecting from a set of models $\mathcal{C}_\Theta(L_{(m)})$ the model nearest to the maximum-likelihood model based on the cost function $\psi(\theta,\hat{\theta})$ and then quantizing the speech given the selected model. As quantization of the predictor parameters corresponds to our model selection, it is then relevant to compare the distortion measure of (14.32) with the distortion measures that are commonly used for the linear-prediction parameters in existing speech coders.

To provide insight, it is useful to write $R_x(\mathrm{e}^{\mathrm{i}\omega}) = R_{\hat{\theta}}(\mathrm{e}^{\mathrm{i}\omega})\, R_w(\mathrm{e}^{\mathrm{i}\omega})$, where $R_w(\mathrm{e}^{\mathrm{i}\omega})$ represents a *remainder* power-spectral density that captures the spectral error of the maximum likelihood model. If the model family is of low order, then $R_w(\mathrm{e}^{\mathrm{i}\omega})$ includes the spectral *fine structure*. We can rewrite (14.32) as

$$\psi(\theta,\hat{\theta}) = \frac{k}{4\pi}\int_0^{2\pi}\left[-\log\left(\frac{R_{\hat{\theta}}(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})}\right) + \left(\frac{R_{\hat{\theta}}(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})} - 1\right) R_w(\mathrm{e}^{\mathrm{i}\omega})\right] \mathrm{d}\omega\,. \quad (14.36)$$

Interestingly, (14.36) reduces to the well-known *Itakura–Saito criterion* [14.42] if $R_w(\mathrm{e}^{\mathrm{i}\omega})$ is set to unity.

It is common (e.g., [14.43]) to relate different criteria through the series expansion $u = 1+\log(u)+\frac{1}{2}[\log(u)]^2+\cdots$. Assuming small differences between the optimal model $\hat{\theta}$ and the model from the set θ, (14.36) can be written

$$\psi(\theta,\hat{\theta}) \cong \frac{k}{4\pi}\int_0^{2\pi}\left\{\left[R_w(\mathrm{e}^{\mathrm{i}\omega})-1\right]\log\left(\frac{R_{\hat{\theta}}(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})}\right) + \frac{1}{2}R_w(\mathrm{e}^{\mathrm{i}\omega})\left[\log\left(\frac{R_{\hat{\theta}}(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})}\right)\right]^2\right\}\mathrm{d}\omega\,. \quad (14.37)$$

Equation (14.37) needs to be accurate only for nearest neighbors of $\hat{\theta}$.

We can simplify (14.37) further. With our assumptions for the autoregressive models, $\mathbf{R}_\theta(z)$ is related to monic minimum-phase polynomials through (14.27) and the further assumption that their gains σ are identical (i. e., is not considered here), this implies that

$$\frac{1}{2\pi}\int_0^{2\pi}\log(\mathbf{R}_\theta)\,\mathrm{d}\omega = \frac{1}{2\pi}\int_0^{2\pi}\log(R_{\hat{\theta}})\,\mathrm{d}\omega = \log(\sigma^2)\,. \quad (14.38)$$

This means that we can rewrite (14.37) as

$$\psi(\theta,\hat{\theta}) \cong \frac{k}{4\pi}\int_0^{2\pi} R_w(\mathrm{e}^{\mathrm{i}\omega})\left\{\log\left(\frac{R_{\hat{\theta}}(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})}\right) + \frac{1}{2}\left[\log\left(\frac{R_{\hat{\theta}}(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})}\right)\right]^2\right\}\mathrm{d}\omega\,. \quad (14.39)$$

Equation (14.39) forms the basic measure that must be optimized for the selection of the model from a set of models, i. e., for the optimal quantization of the model parameters.

If we can neglect the impact of $R_w(z)$, then (using the result of (14.38)) minimizing (14.39) is equivalent to minimizing

$$\psi(\theta,\hat{\theta}) \cong \frac{k}{8\pi}\int_0^{2\pi}\left[\log\left(\frac{R_{\hat{\theta}}(\mathrm{e}^{\mathrm{i}\omega})}{\mathbf{R}_\theta(\mathrm{e}^{\mathrm{i}\omega})}\right)\right]^2\mathrm{d}\omega\,, \quad (14.40)$$

which is the well-known *mean squared log spectral distortion*, scaled by the factor $k/4$. Except for this scaling factor, (14.39) is precisely the criterion that is commonly used (e.g., [14.35, 44, 45]) to evaluate performance of quantizers for autoregressive (AR) model parameters. This is not unreasonable as the neglected modeling error $R_w(\mathrm{e}^{\mathrm{i}\omega})$ is likely uncorrelated with the model quantization error.

14.4.3 Bit Allocation for the Model

The AR model is usually described with a small number of parameters (as mentioned, $p = 10$ is common for 8 kHz sampling rate). Thus, the spectral data must lie on a manifold of dimension p or less in the log spectrum space. At high bit allocations, where measurement noise dominates (see also the end of Sect. 14.3.3), the manifold dimension is p and the spectral distortion is expected to scale as

$$D(L_{(m)}) = \frac{k}{4}\beta^2 N_{\mathrm{AR}}^{-\frac{2}{p}} = \frac{k}{4}\beta^2\,\mathrm{e}^{-\frac{2}{p}L_{(m)}}\,, \quad (14.41)$$

where β is a constant and N_{AR} is the number of spectral models in the family and $L_{(m)} = \log(N_{\mathrm{AR}})$. At higher spectral distortion levels, it has been observed that the physics of the vocal tract constrains the dimensionality of the manifold. This means that (14.41) is replaced by

$$D(L_{(m)}) = \frac{k}{4}\beta^2\,\mathrm{e}^{-\frac{2}{\kappa}L_{(m)}} \quad (14.42)$$

with $\kappa < p$. This behavior was observed for trained codebooks over a large range in [14.44] (similar behavior was

observed for cepstral parameters in [14.46]) and for specific vowels in [14.47]. The results of [14.44] correspond to $\kappa = 7.1$ and $\beta = 0.80$.

The mean of (14.31) becomes

$$\mathrm{E}[\eta] = L_{(m)} + \frac{k}{4}\beta^2 \mathrm{e}^{-\frac{2}{\kappa}L_{(m)}} \,. \tag{14.43}$$

Differentiating towards $L_{(m)}$ we find that the optimal bit allocation for the AR model selection to be

$$L_{(m)} = \frac{\kappa}{2}\log\left(\beta^2\frac{k}{2\kappa}\right) \,, \tag{14.44}$$

which is logarithmically dependent on k. Using the observed data of [14.44], we obtain an optimal rate of about 17 bits for 8 kHz sampled speech at a 20 ms block size. The corresponding mean spectral distortion is about 1.3 dB. The distortion is similar to the mean estimation errors found in experiments on linear predictive methods on speech sounds [14.48].

The 17 bit requirement for the prediction parameter quantizer is similar to that obtained by the best available prediction parameter quantizers that operate on single blocks and bounds obtained for these methods [14.35, 49–52]. In these systems the lowest bit allocation for 20 ms blocks is about 20 bits. However, the performance of these coders is entirely based on the often quoted 1 dB threshold for transparency [14.35]. The definition of this empirical threshold is consistent with the conventional two-step approach: the model parameters are first quantized using a separately defined criterion, and the speech signal is quantized thereafter based on a weighted squared error criterion. In contrast, we have shown that a single distortion measure operating on the speech vector suffices for this purpose.

We conclude that the definition of a squared-error criterion for the speech signal leads to a bit allocation for the autoregressive model. No need exists to introduce perception based thresholds on log spectral distortion.

14.4.4 Remarks on Practical Coding

The two-stage approach is standard practice in linear-prediction-based (autoregressive-model-based) speech codecs. In the selection stage, weighted squared error criteria in the so-called line-spectral frequency (LSF) representation of the prediction parameters are commonly used, e.g., [14.34–37]. If the proper weighting is used, then the criterion can be made to match the log spectral distortion measure [14.53] that we derived above.

The second stage is the selection of a speech codebook entry from a codebook corresponding to the selected model. The separation into a set of models simplifies this selection. In general, this means that a speech-domain codebook must be available for each model. It was recently shown that the computational or storage requirements for optimal speech-domain codebooks can be made reasonable by using a single codebook for each set of speech sounds that are similar except for a unitary transform [14.54]. The method takes advantage of the fact that different speech sounds may have similar statistics after a suitable unitary transform and can, therefore, share a codebook. As the unitary transform does not affect the Euclidian distance, it also does not affect the optimality of the codebook.

In the majority of codecs the speech codebooks are generated in real time, with the help of the model obtained in the first stage. This approach is the so-called *analysis-by-synthesis* approach. It can be interpreted as a method that requires the *synthesis* of candidate speech vectors (our speech codebook), hence the name. Particularly common is the usage of the analysis-by-synthesis approach for the autoregressive model [14.16,55]. While the analysis-by-synthesis approach has proven its merit and is used in hundreds of millions of communication devices, it is not optimal. It was pointed out in [14.54] that analysis-by-synthesis coding inherently results in a speech-domain codebook with quantization cells that have a suboptimal shape, limiting performance.

14.5 Distortion Measures and Coding Architecture

An objective of coding is the removal of irrelevancy. This means that precision is lost and that we introduce a difference between the original and the decoded signal, the error signal. So far we have considered basic quantization theory and how modeling can be introduced in this quantization structure. We based our discussion on a mean squared error distortion measure for the speech vector. As discussed in Sect. 14.2.2, the proper measure is the decrease in signal quality as perceived by human listeners. That is, the goal in speech coding is to minimize the perceived degradation resulting from an encoding at a particular rate. This section discusses

methods for integrating perceptually motivated criteria into a coding structure.

To base coding on perceived quality degradation, we must define an appropriate quantitative measure of the perceived distortion. Reasonable objectives for a good distortion measure for a speech codec are a good prediction of experimental data on human perception, mathematical tractability, low delay, and low computational requirements.

A major aspect in the definition of the criterion is the representation of the speech signal the distortion measure operates on. Most straightforward is to quantize the speech signal itself and use the distortion measure as a selection criterion for code vectors and as a means to design the quantizers. This coding structure is commonly used in speech coders based on linear-predictive coding. An alternative coding structure is to apply a transform towards a domain that facilitates a simple distortion criterion. Thus, in this approach, we first perform a mapping to a *perceptual domain* (preprocessing) and then quantize the mapped signal in that domain. At the decoder we apply the inverse mapping (postprocessing). This second architecture is common in transform coders aimed at encoding audio signals at high fidelity.

We start this section with a subsection discussing the squared error criterion, which is commonly used because of its mathematical simplicity. In Subsects. 14.5.2 and 14.5.3 we then discuss models of perception and how the squared error criterion can be used to represent these models. We end the section with a subsection discussing in some more detail the various coding architectures.

14.5.1 Squared Error

The squared-error criterion is commonly used in coding, often without proper physical motivation. Such usage results directly from its mathematical tractability. Given a data sequence, optimization of the model parameters for a model family often leads to a set of linear equations that is easily solved.

For the k-dimensional speech vector $\boldsymbol{x}^k$, the basic squared-error criterion is

$$\eta = (\boldsymbol{x}^k - \hat{\boldsymbol{x}}^k)^{\mathrm{H}}(\boldsymbol{x}^k - \hat{\boldsymbol{x}}^k)\,, \tag{14.45}$$

where the superscript 'H' denotes the Hermitian conjugate and $\hat{\boldsymbol{x}}^k$ is the reconstruction vector upon encoding and decoding. Equation (14.45) quantifies the variance of the signal error. Unfortunately, variance cannot be equated to loudness, which is the psychological correlate of variance. At most we can expect that, for a given original signal, a scaling of the error signal leads to a positive correlation between perceived distortion and squared error.

While the squared error in its basic form is not representative of human perception, adaptive weighting of the squared-error criterion can lead to improved correspondence. By means of weighting we can generalize the squared-error criterion to a form that allows inclusion of knowledge of perception (the formulation of the weighted squared error criterion for a specific perceptual model is described in Subsects. 14.5.2 and 14.5.3). To allow the introduction of perceptual effects, we linearly weight the error vector $\boldsymbol{x}^k - \hat{\boldsymbol{x}}^k$ and obtain

$$\eta = (\boldsymbol{x}^k - \hat{\boldsymbol{x}}^k)^{\mathrm{H}}\mathbf{H}^{\mathrm{H}}\mathbf{H}(\boldsymbol{x}^k - \hat{\boldsymbol{x}}^k)\,, \tag{14.46}$$

where $\mathbf{H}$ is an $m \times k$ matrix, where m depends on the weighting invoked. As we will see below, many different models of perception can be approximated with the simple weighted squared-error criterion of (14.46). In general, the weighting matrix $\mathbf{H}$ adapts to x^k, that is $\mathbf{H}(\boldsymbol{x}^k)$ and

$$\boldsymbol{y}^m = \mathbf{H}(\boldsymbol{x}^k)\,\boldsymbol{x}^k \tag{14.47}$$

can be interpreted as a perceptual-domain representation of the signal vector for a region of $\boldsymbol{x}^k$ where $\mathbf{H}(\boldsymbol{x}^k)$ is approximately constant.

The inclusion of the matrix $\mathbf{H}$ in the formulation of the squared-error criterion generally results in a significantly higher computational complexity for the evaluation of the criterion. Perhaps more importantly, when the weighted criterion of (14.46) is adaptive, then the optimal distribution of the code vectors (Sect. 14.3.3) for constrained-entropy coding is no longer uniform in the speech domain. This has significant implications for the computational effort of a coding system.

The formulation of (14.46) is commonly used in coders that are based on an autoregressive model family, i. e., linear-prediction-based analysis-by-synthesis coding [14.16]. (The matrix $\mathbf{H}$ then usually includes the autoregressive model, as the speech codebook is defined as a filtering of an excitation codebook.) Also in the context of this class of coders, the vector $\mathbf{H}\boldsymbol{x}^k$ can be interpreted as a perceptual-domain vector. However, because $\mathbf{H}$ is a function of $\boldsymbol{x}^k$ it is not straightforward to define a codebook in this domain.

The perceptual weighting matrix $\mathbf{H}$ often represents a filter operation. For a filter with impulse response $[h_0, h_1, h_2, \cdots]$, the matrix $\mathbf{H}$ has a Toeplitz structure:

$$\mathbf{H} = \begin{pmatrix} h_0 & 0 & \cdots \\ h_1 & h_0 & \cdots \\ h_2 & h_1 & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix} . \tag{14.48}$$

For computational reasons, it may be convenient to make the matrix $\mathbf{H}^{\mathrm{H}}\mathbf{H}$ Toeplitz. If the impulse response has time support p then $\mathbf{H}^{\mathrm{H}}\mathbf{H}$ is Toeplitz if $\mathbf{H}$ is selected to have dimension $(m+p)\times m$ [14.19].

Let us consider how the impulse response $[h_0, h_1, h_2, \cdots]$ of (14.48) is typically constructed for the case of linear-predictive coding. The impulse response is constructed from the signal model. Let the transfer function of the corresponding autoregressive model be, as in (14.26)

$$A(z)^{-1} = \frac{\sigma}{1 + a_1 z^{-1} \cdots + a_p z^{-p}} , \tag{14.49}$$

where the a_i are the prediction parameters and σ is the gain. A weighting that is relatively flexible and has low computational complexity is then [14.56]

$$H(z) = \frac{A(z/\gamma_1)}{A(z/\gamma_2)} , \tag{14.50}$$

where γ_1 and γ_2 are parameters that are selected to accurately describe the impact of the distortion on perception. The sequence $[h_0, h_1, h_2, \cdots]$ of (14.48) is now simply the impulse response of $H(z)$. The parameters γ_1 and γ_2 are selected to approximate perception where $1 \geq \gamma_1 > \gamma_2 > 0$. The filter $A(z/\gamma_1)$ deemphasizes the envelope of the power spectral density, which corresponds to decreasing the importance of spectral peaks. The filter $1/A(z/\gamma_2)$ undoes some of this emphasis for a smoothed version of the spectral envelope. The effect is roughly that $1/A(z/\gamma_2)$ limits the spectral reach of the deemphasis $A(z/\gamma_1)$. In other words, the deemphasis of the spectrum is made into a local effect.

To understand coders of the transform model family, it is useful to interpret (14.46) in the frequency domain. We write the discrete Fourier transform (DFT) as the unitary matrix $\mathbf{F}$ and the define a frequency-domain weighting matrix $\mathbf{W}$ such that

$$\mathbf{H}\boldsymbol{x}^k = \mathbf{F}^{\mathrm{H}}\mathbf{W}\mathbf{F}\boldsymbol{x}^k , \tag{14.51}$$

The matrix $\mathbf{W}$ provides a weighting of the frequency-domain vector $\mathbf{F}\boldsymbol{x}^k$. If, for the purpose of our discussion, we neglect the difference between circular and linear convolution and if $\mathbf{H}$ represents a filtering operation (convolution) as in (14.48), then $\mathbf{W}$ is diagonal. To account for perception, we must adapt $\mathbf{W}$ to the input vector $\boldsymbol{x}^k$ (or equivalently, to the frequency-domain vector $\mathbf{F}\boldsymbol{x}^k$) and it becomes a function $\mathbf{W}(\boldsymbol{x}^k)$: Equation (14.50) could be used as a particular mechanism for such weighting. However, in the transform coding context, so-called *masking* methods, which are described in Sect. 14.5.2, are typically used to find $W(\boldsymbol{x}^k)$.

As mentioned before, the random vector $\boldsymbol{Y}^m = \mathbf{H}\boldsymbol{X}^k$ (or, equivalently, the vector $\mathbf{W}\mathbf{F}\boldsymbol{X}^k$) can be considered as a perceptual-domain description. Assuming smooth behavior of $\mathbf{H}(\boldsymbol{x}^k)$ as a function of $\boldsymbol{x}^k$, this domain can then be used as the domain for coding. A codebook must be defined for the perceptual-domain vector $\boldsymbol{Y}^m$ and we select entries from this codebook with the unweighted squared error criterion. This approach is common in transform coding. When this coding in the perceptual domain is used, the distortion measure does not vary with the vector $\boldsymbol{y}^m$, and a uniform quantizer is optimal for $\boldsymbol{Y}^m$ for the constrained-entropy case. If the mapping to the perceptual domain is unique and invertible (which is not guaranteed by the formulation), then $\boldsymbol{y}^m = \mathbf{H}(\boldsymbol{x}^k)\boldsymbol{x}^k$ ensures that $\boldsymbol{x}^k$ is specified when $\boldsymbol{y}^m$ is known and only indices to the codebook for $\boldsymbol{Y}^m$ need to be encoded. In practice, the inverse mapping may not be unique, resulting in problems at block boundaries and the inverse may be difficult to compute. As a result it is common practice to quantize and transmit the weighting $\mathbf{W}$, e.g., [14.57, 58].

14.5.2 Masking Models and Squared Error

Extensive quantitative knowledge of auditory perception exists and much of the literature on quantitative descriptions of auditory perception relates to the concept of *masking*, e.g., [14.59–63]. The masking-based description of the operation of the auditory periphery can be used to include the effect of auditory perception in speech and audio coding. Let us define an arbitrary signal that we call the *masker*. The masker implies a set of second signals, called *maskees*, which are defined as signals that are not audible when presented in the presence of the masker. That is, the maskee is below the *masking threshold*. Masking explains, for example, why a radio must be made louder in a noisy environment such as a car. We can think of masking as being

a manifestation of the internal precision of the auditory periphery.

In general, laws for the masking threshold are based on psychoacoustic measurements for the masker and maskee signals that are constructed independently and then added. However, it is clear that the coding error is correlated to the original signal. In the context of masking it is a commonly overlooked fact that, for ideal coding, the coding error signal is, under certain common conditions, independent of the reconstructed signal [14.12]. Thus, a reasonable objective of audio and speech coding is to ensure that the coding error signal is below the masking threshold of the *reconstructed* signal.

Masking is quantified in terms of a so-called masking curve. We provide a generalized definition of such a curve. Let us consider a signal vector $\boldsymbol{x}^k$ with k samples that is defined in $\mathbb{R}^k$. We define a perceived-error measurement domain by any invertible mapping $\mathbb{R}^k \to \mathbb{R}^m$. Let $\{\boldsymbol{e}_i^m\}_{i\in\{0,\cdots,m-1\}}$ be the unit-length basis vectors that span $\mathbb{R}^m$. We then define the m-dimensional *masking curve* as [14.64] JND_i, $i \in 0, \cdots, m-1$, where the scalar JND_i is the *just-noticeable difference* (JND) for the basis vector $\boldsymbol{e}_i^m$. That is, the vector $JND_i\,\boldsymbol{e}_i^m$ is precisely at the threshold of being audible for the given signal vector.

Examples of the masking curve can be observed in the time and the frequency domain. The frequency-domain representation of $\boldsymbol{x}^k$ is $\mathbf{F}\boldsymbol{x}^k$. *Simultaneous* masking is defined as the masking curve for $\mathbf{F}\boldsymbol{x}^k$, i. e., the just-noticeable amplitudes for the frequency unit vectors $\boldsymbol{e}_1$, $\boldsymbol{e}_2$, etc. In the time domain we refer to *nonsimultaneous* (or *forward* and *backward*) masking depending on whether the time index i of the unit vectors $\boldsymbol{e}_i$ is prior to or after the main event in the masker (e.g., an onset). Both the time-domain (temporal) and frequency-domain masking curves are asymmetric and dependent on the loudness of the masker. A loud sound leads to a rapid decrease in auditory acuity, followed by a slow recovery to the default level. The recovery may take several hundreds of ms and causes forward masking. The decrease in auditory acuity before a loud sound, backward masking, extends only over very short durations (at most a few ms). Similar asymmetry occurs in the frequency domain, i. e., in simultaneous masking. Let us consider a tone. The auditory acuity is decreased mostly at frequencies higher than the tone. The acuity increases more rapidly from the masker when moving towards lower frequencies than when moving towards higher frequencies, which is related to the decrease in frequency resolution with increasing frequency. A significant difference exists in the masking between tonal and noise-like signals. We refer to [14.63, 65, 66] for further information on masking.

The usage of masking is particularly useful for coding in the perceptual domain with a constraint that the quality is to be transparent (at least according to the perceptual knowledge provided). For example, consider a transform coder (based on either the discrete cosine transform or the DFT). In this case, the quantization step size can be set to be the JND as provided by the simultaneous masking curve [14.57].

Coders are commonly subject to a bit-rate constraint, which means knowledge of the masking curve is not sufficient. A distortion criterion must be defined based on the perceptual knowledge given. A common strategy in audio coding to account for simultaneous masking is to use a weighted squared error criterion, with a diagonal weighting matrix $\mathbf{H}$ that is reciprocal of the masking threshold [14.27, 58, 67–70]. In fact, this is a general approach that is useful to convert a masking curve in any measurement domain:

$$\mathbf{H} = \begin{pmatrix} \frac{1}{JND_1} & 0 & \cdots \\ 0 & \frac{1}{JND_2} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix} , \tag{14.52}$$

where it is understood that the weighting matrix $\mathbf{H}$ is defined in the measurement domain. To see this consider the effect of the error vector $JND_i\,\boldsymbol{e}_i^m$ on the squared error:

$$\begin{aligned} \eta &= JND_i^2\,\boldsymbol{e}_i^{m\mathrm{H}}\mathbf{H}^{\mathrm{H}}\mathbf{H}\boldsymbol{e}_i^m \\ &= JND_i^2\;JND_i^{-2} = 1 \,. \end{aligned} \tag{14.53}$$

Thus, the points on the masking curve are defined as the amplitudes of basis vectors that lead to a unit distortion. This is a reasonable motivation for the commonly used reciprocal-weighting approach for the squared-error criterion defined by the weighting described in (14.52). However, it should be noted that for this formulation the distortion measure does not vanish below the masking threshold. A more-complex approach where the distortion measure does vanish below the JND is given in [14.71].

14.5.3 Auditory Models and Squared Error

The weighting procedure of (14.52) (possibly in combination with the transform to the measurement domain)

is an operation that transforms the signal to a perceptually relevant domain. Thus, the operation can be interpreted as a simple auditory model. Sophisticated models of the auditory periphery that directly predict the input to the auditory nerve have also been developed, e.g., [14.72–77]. Despite the existence of such quantitative models of perception, their application in speech coding has been limited. Only a few examples [14.78,79] of the explicit usage of existing quantitative knowledge of auditory perception in speech coding speech exist. In contrast, in the field of audio coding the usage of quantitative auditory knowledge is common. Transform coders can be interpreted as methods that perform coding in the perceptual domain, using a simple perceptual model, usually based on (simultaneous) masking results.

We can identify a number of likely causes for the lack of usage of auditory knowledge in speech coding. First, the structure of speech coders and the constraint on computational complexity naturally leads to speech-coding-specific models of auditory perception, such as (14.50). The parameters of these simple speech-coding-based models are optimized directly based on coding performance. Second, the perception of the periodicity nature of voiced speech, often referred to as the perception of *pitch*, is not well understood in a quantitative manner. It is precisely the distortion associated with the near-periodic nature of voiced speech that is often critical for the perceived quality of the reconstructed signal. An argument against using a quantitative model based on just-noticeable differences (JNDs) is that JNDs are often exceeded significantly in speech coding. While the weighting of (14.52) is reasonable near the JND threshold value, it may not be accurate in the actual operating region of the speech coder. Major drawbacks of using sophisticated models based on knowledge of the auditory periphery are that they tend to be computationally expensive, have significant latency, and often lead to a representation that has many more dimensions than the input signal. Moreover the complexity of the model structure makes inversion difficult, although not impossible [14.80].

The complex structure of auditory models that describe the functionality of the auditory periphery is time invariant. We can replace it by a much simpler structure at the cost of making it time variant. That is, the mapping from the speech domain to the perceptual domain can be simplified by approximating this mapping as locally linear [14.79]. Such an approximation leads to the *sensitivity matrix approach*, which was first introduced in a different context by [14.53] and described in a rigorous general manner in [14.81]. If a mapping from the speech domain vector $\boldsymbol{x}^k$ to an auditory domain vector $\boldsymbol{y}^m$ (as associated with a particular model of the auditory periphery) can be approximated as locally linear, then for a small coding error $\boldsymbol{x}^k - \hat{\boldsymbol{x}}^k$, we can write $\boldsymbol{y}^m - \hat{\boldsymbol{y}}^m$ as a matrix multiplication of $\boldsymbol{x}^k - \hat{\boldsymbol{x}}^k$:

$$\boldsymbol{y}^m - \hat{\boldsymbol{y}}^m \approx \mathbf{H}\left(\boldsymbol{x}^k - \hat{\boldsymbol{x}}^k\right), \tag{14.54}$$

where $\mathbf{H}$ is an $m \times k$ matrix. This means that, for the set of codebook vectors from $C_{\boldsymbol{X}^k}$ that is sufficiently close to $\boldsymbol{x}^k$, the squared-error criterion of (14.46) forms an approximation to the psychoacoustic measure. Moreover, selecting the nearest codebook entry from $C_{\boldsymbol{X}^k}$ using (14.46) results in the globally optimal codebook vector for the input vector $\boldsymbol{x}^k$. In the sensitivity matrix approach, the first step for each speech vector $\boldsymbol{x}^k$ is to find the $k \times k$ *sensitivity matrix* $\mathbf{H}^{\mathrm{T}}\mathbf{H}$. This operation is based on an analysis of the distortion criterion [14.79]. Once this has been done, the selection of the codebook entries is similar to that for a signal-invariant weighted squared-error distortion measure.

In the sensitivity matrix approach, the matrix $\mathbf{H}$ is a function of the past and future signal:

$$\mathbf{H} = \mathbf{H}(\cdots, \boldsymbol{x}_{i-1}^k, \boldsymbol{x}_i^k, \boldsymbol{x}_{i+1}^k, \cdots), \tag{14.55}$$

where $\boldsymbol{x}_i^k$ is the current speech vector, $\boldsymbol{x}_{i-1}^k$ is the previous speech vector, etc. To avoid the introduction of latency, the future speech vectors can be replaced by a prediction of these vectors from the present and past speech vectors.

The sensitivity matrix approach requires that the mapping from speech domain to perceptual domain is continuous and differentiable, which is not the case for psychoacoustic models. The approximation of such discontinuities by continuous functions generally leads to satisfactory results.

The sensitivity matrix approach is well motivated in the context of a speech-domain codebook $C_{\boldsymbol{X}^k}$ and a criterion that consists of a perceptual transform followed by the squared error criterion. The search through the speech-domain codebook with a perceptual criterion then reduces to searching with a weighted squared-error criterion.

The benefit of the sensitivity matrix approach is not so obvious if the signal vector codebook, $C_{\boldsymbol{X}^k}$, is defined in the perceptual domain. However, it can be useful if the perceptual transform is known to the decoder (by, for example, transmission of an index). The perceptual domain often has higher dimensionality than the corresponding

speech block. The singular-value decomposition of $\mathbf{H}$ can then be used to reduce the dimensionality of the perceptual domain error vector to k. Let $\mathbf{H} = \mathbf{VDU}$ be a singular value decomposition, where $\mathbf{V}$ is an $m \times m$ unitary matrix, $\mathbf{D}$ is a $m \times k$ diagonal matrix and $\mathbf{U}$ is a unitary $k \times k$ matrix

14.5.4 Distortion Measure and Coding Architecture

As we have seen, the distortion measure has a significant impact on the architecture of a speech coder. In this subsection, we summarize the above discussion from a codec-architecture viewpoint.

Speech-Domain Codebook

The most straightforward architecture for a speech coder is to define the codebook in the speech domain and use an appropriate distortion measure during encoding and during training of the codebook. In general, the measure is adaptive. This approach, which is shown in Fig. 14.1, is most common in speech coding. An advantage is that the decoder does not require knowledge of the time-varying distortion measure.

We saw in Sect. 14.3.4 that it can be advantageous to use speech codebooks that are associated with models. This simplifies the codebook structure and, in the case of constrained-resolution coding, its size. The codebooks can be generated in real time as, for example, in the case of linear-prediction (autoregressive model)-based analysis-by-synthesis coding [14.16, 55].

The underlying aim of the speech-domain codebook architecture is generally to approximate auditory perception by an adaptively weighted squared-error criterion. Usually, this criterion is heuristic and based on tuning within the context of the coder (as is typically done for (14.50)). However, as shown in Sect. 14.5.3, the sensitivity analysis method facilitates the usage of complex auditory models. This approach requires, at least in principle, no further tuning.

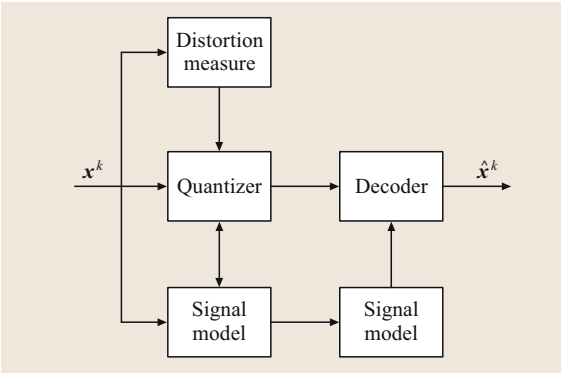


Fig. 14.1 Common architecture for coding with a distortion measure. A signal quantization index and a signal model index are transmitted

Disadvantages of the weighted squared-error criterion are its computational complexity and that, in contrast to the unweighted squared-error measure, it does not lead to uniform codebooks for the constrained-entropy case. In the context of autoregressive-model-based analysis-by-synthesis coding, many procedures have been developed to reduce the computational complexity of the weighted squared-error criterion [14.18, 19, 82].

Perceptual-Domain Codebook

As an alternative to defining the codebook in the speech domain, we can define the codebook in a perceptual domain, as is shown in Fig. 14.2. We define a perceptual domain as a domain where the unweighted squared error criterion can be applied. The most elegant paradigm requires no information about the speech vector other than its index in the perceptual domain codebook. This elegance applies if the mapping to the perceptual domain is injective (one to one). Then if $\boldsymbol{y}^m$ is known, $\boldsymbol{x}^k$ is also known. An example is an auditory model that is a weighted DFT or DCT with a one-to-one function $\mathbb{R}^k \to \mathbb{R}^k$ that maps $\boldsymbol{x}^k$ into $\boldsymbol{y}^k$. An inverse func-

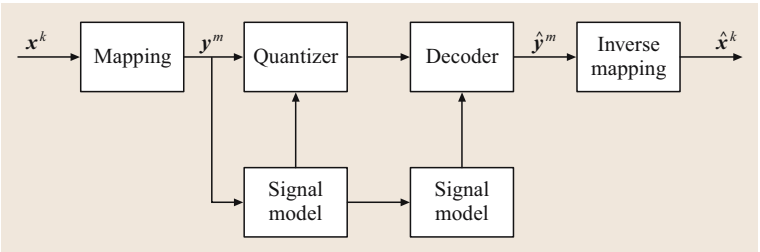


Fig. 14.2 Ideal architecture for coding in the perceptual domain, with invertible mapping. A perceptual-domain quantization index and a signal model index are transmitted. The signal model can be omitted

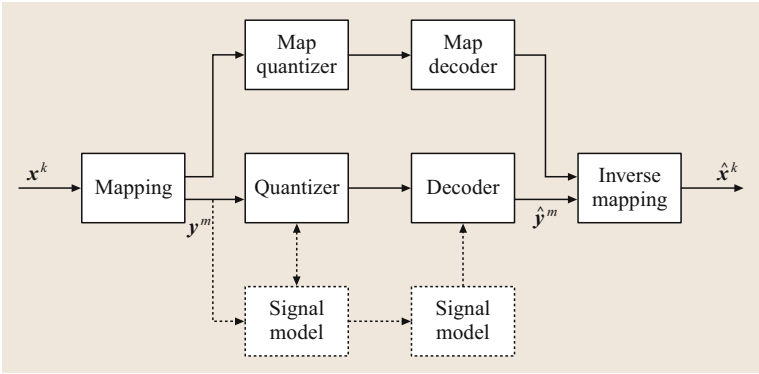


Fig. 14.3 Architecture for coding in the perceptual domain with encoded mapping. This architecture is common in transform coding. A mapping index, a perceptual-domain quantization index, and a signal model index are transmitted. The signal model is commonly omitted

tion can be derived by the decoder from the quantized vector $\hat{\boldsymbol{y}}^{\mathrm{k}}$.

The non-uniqueness of the auditory mapping from the speech domain to the auditory domain results in practical problems. Particularly if the models are not accurate, the nonuniqueness can result in mismatches between coding blocks and severe audible distortion.

The uniqueness issue for the mapping can be solved, at the cost of increased rate, by transmitting information about the mapping as is shown in Fig. 14.3. For example, for transform coders used for audio signals, the masking curve is commonly transmitted, e.g., [14.52, 58, 67–69]. To reduce the rate required, this is commonly done on a per-frequency-band basis. The bands generally are uniformly spaced on an equivalent rectangular bandwidth (ERB) or mel scale. In practice the masking curve is not always transmitted directly, but a maximum amplitude, and the number of quantization levels for each band are transmitted, e.g., [14.27, 68, 70].

14.6 Summary

This chapter discussed the principles underlying the transmission of speech (and audio) signals. The main attributes of coding, rate, quality, robustness to channel errors, delay, and computational complexity were discussed first. We then provided a generic perspective of speech coding.

Each block of speech samples was described as a random vector. Information about the vector was transmitted in the form of a codebook index. The relation between the reconstruction vector density and the data density was given. We then modeled the probability density of the speech vector as a weighted sum of component probability density functions, each describing the speech vector probability density for a particular speech sound. Each such component density function corresponds to a *model*. In the case of linear-prediction-based (equivalent to autoregressive-model-based) coding, each component function (and thus model) is characterized by a set of predictor parameters. The approach results in the standard two-step speech coding approach, in which we first extract the model and then code the speech vector given the model. We showed that this approach leads to standard distortion measures used for the quantization of the predictor parameters.

We showed that the number of models (component probability density functions) is independent of the overall coding rate. Thus, the rate spent on linear-prediction parameters in linear-predictive coding should not vary with rate (at least when the rate is high). It was shown that practical coders indeed have this behavior. We emphasized also that the rate allocated to the model (the predictor parameters) is not a direct function of a perceptual threshold, but the result of an optimal trade-off between the rate allocated for the speech given the model and the rate allocated for the model. We showed that it is possible to calculate the rate allocation for the model (the bit allocation for the predictor parameters) and that the result provided is close to practical codec configurations. We discussed analysis-by-synthesis coding as a particular application of the two-stage coding method. We noted that analysis-by-synthesis coding is not optimal because the speech-domain codebook has suboptimal quantization cell shapes.

Finally we discussed how perception can be integrated into the coding structure. We distinguished coding in a perceptually relevant domain, which is commonly used in audio coding, from coding in the speech-signal domain, which is commonly used in speech coding. The advantage of coding in the perceptual domain is that a simple squared-error criterion can be used. However, in practice the method generally requires some form of encoding of the mapping, so that the decoder can perform an inverse mapping. Improved mapping procedures may change this requirement.

References

14.1 W.B. Kleijn, K.K. Paliwal: An introduction to speech coding. In: *Speech Coding and Synthesis*, ed. by W.B. Kleijn, K.K. Paliwal (Elsevier, Amsterdam 1995) pp. 1–47

14.2 R.V. Cox: Speech coding standards. In: *Speech Coding and Synthesis*, ed. by W.B. Kleijn, K.K. Paliwal (Elsevier, Amsterdam 1995) pp. 49–78

14.3 R. Salami, C. Laflamme, J. Adoul, A. Kataoka, S. HAyashi, T. Moriya, C. Lamblin, D. Massaloux, S. Proust, P. Kroon, Y. Shoham: Design and description of CS-ACELP: a toll quality 8 kb/s speech coder, IEEE Trans. Speech Audio Process. **6**(2), 116–130 (1998)

14.4 B. Bessette, R. Salami, R. Lefebvre, M. Jelinek, J. Rotola-Pukkila, J. Vainio, H. Mikkola: The adaptive multirate wideband speech codec (amr-wb), IEEE Trans. Speech Audio Process. **6**(8), 620–636 (2002)

14.5 ITU-T Rec. P.800: *Methods for Subjective Determination of Transmission Quality* (1996)

14.6 A.W. Rix: Perceptual speech quality assessment – a review, Proc. IEEE ICASSP, Vol. 3 (2004) pp. 1056–1059

14.7 S. Möller: *Assessment and Prediction of Speech Quality in Telecommunications* (Kluwer Academic, Boston 2000)

14.8 P. Kroon: Evaluation of speech coders. In: *Speech Coding and Synthesis*, ed. by W.B. Kleijn, K.K. Paliwal (Elsevier, Amsterdam 1995) pp. 467–493

14.9 W. Stallings: *High-speed networks: TCP/IP and ATM design principles* (Prentice Hall, Englewood Cliffs 1998)

14.10 Information Sciences Institute: Transmission control protocol, IETF RFC793 (1981)

14.11 J. Postel: User datagram protocol, IETF RFC768 (1980)

14.12 T.M. Cover, J.A. Thomas: *Elements of Information Theory* (Wiley, New York 1991)

14.13 N. Kitawaki, K. Itoh: Pure delay effects on speech quality in telecommunications, IEEE J. Sel. Area. Comm. **9**(4), 586–593 (1991)

14.14 J. Cox: The minimum detectable delay of speech and music, Proc. IEEE ICASSP, Vol. 1 (1984) pp. 136–139

14.15 J. Chen: A robust low-delay CELP speech coder at 16 kb/s. In: *Advances in Speech Coding*, ed. by B.S. Atal, V. Cuperman, A. Gersho (Kluwer Academic, Dordrecht 1991) pp. 25–35

14.16 B.S. Atal, M.R. Schroeder: Stochastic coding of speech at very low bit rates, Proc. Int. Conf. Comm. (1984) pp. 1610–1613

14.17 J.-P. Adoul, P. Mabilleau, M. Delprat, S. Morisette: Fast CELP coding based on algebraic codes, Proc. IEEE ICASSP (1987) pp. 1957–1960

14.18 I.M. Trancoso, B.S. Atal: Efficient procedures for selecting the optimum innovation in stochastic coders, IEEE Trans. Acoust. Speech **38**(3), 385–396 (1990)

14.19 W.B. Kleijn, D.J. Krasinski, R.H. Ketchum: Fast methods for the CELP speech coding algorithm, IEEE Trans. Acoust. Speech **38**(8), 1330–1342 (1990)

14.20 T. Lookabough, R. Gray: High-resolution theory and the vector quantizer advantage, IEEE Trans. Inform. Theory **IT-35**(5), 1020–1033 (1989)

14.21 S. Na, D. Neuhoff: Bennett's integral for vector quantizers, IEEE Trans. Inform. Theory **41**(4), 886–900 (1995)

14.22 S.P. Lloyd: Least squares quantization in PCM, IEEE Trans. Inform. Theory **IT-28**, 129–137 (1982)

14.23 Y. Linde, A. Buzo, R.M. Gray: An algorithm for vector quantizer design, IEEE Trans. Commun. **COM-28**, 84–95 (1980)

14.24 P. Chou, T. Lookabough, R. Gray: Entropy-constrained vector quantization, IEEE Trans. Acoust. Speech **38**(1), 31–42 (1989)

14.25 A. Gersho: Asymptotically optimal block quantization, IEEE Trans. Inform. Theory **25**, 373–380 (1979)

14.26 P. Swaszek, T. Ku: Asymptotic performance of unrestricted polar quantizers, IEEE Trans. Inform. Theory **32**(2), 330–333 (1986)

14.27 R. Vafin, W.B. Kleijn: Entropy-constrained polar quantization and its application to audio coding, IEEE Trans. Speech Audio Process. **13**(2), 220–232 (2005)

14.28 J.J. Rissanen, G. Langdon: Arithmetic coding, IBM J. Res. Devel. **23**(2), 149–162 (1979)

14.29 J. Rissanen: Modeling by the shortest data description, Automatica **14**, 465–471 (1978)

14.30 J. Rissanen: A universal prior for integers and estimation by minimum description length, Ann. Stat. **11**(2), 416–431 (1983)

14.31 P. Grunwald: A tutorial introduction to the minimum description length principle. In: *Advances in Minimum Description Length: Theory and Applications*, ed. by P. Grunwald, I.J. Myung, M. Pitt (MIT, Boston 2005)

14.32 A. Barron, T.M. Cover: Minimum complexity density estimation, IEEE Trans. Inform. Theory **37**(4), 1034–1054 (1991)

14.33 A.H. Gray, J.D. Markel: Distance measures for speech process, IEEE Trans. Acoust. Speech Signal Process. **ASSP-24**(5), 380–391 (1976)

14.34 R. Hagen, P. Hedelin: Low bit-rate spectral coding in CELP a new LSP method, Proc. IEEE ICASSP (1990) pp. 189–192

14.35 K.K. Paliwal, B.S. Atal: Efficient vector quantization of LPC parameters at 24 bits/frame, IEEE Trans. Speech Audio Process. **1**(1), 3–14 (1993)

14.36 C. Xydeas, C. Papanastasiou: Split matrix quantization of lpc parameters, IEEE Trans. Speech Audio Process. **7**(2), 113–125 (1999)

14.37 A. Subramaniam, B. Rao: Speech LSF quantization with rate independent complexity, bit scalability, and learning, Proc. IEEE ICASSP (2001) pp. 705–708

14.38 U. Grenander, G. Szego: *Toeplitz Forms and their Applications* (Chelsea, New York 1984)

14.39 F. Itakura, S. Saito: Speech information compression based on the maximum likelihood estimation, J. Acoust. Soc. Jpn. **27**(9), 463 (1971)

14.40 S. Saito, K. Nakata: *Fundamentals of Speech Signal Process* (Academic, New York 1985)

14.41 P.J. Brockwell, R.A. Davis: *Time Series: Theory and Methods* (Springer, New York 1996)

14.42 F. Itakura, S. Saito: Analysis Synthesis Telephony Based Upon the Maximum Likelihood Method, Reports of 6th Int. Cong. Acoust.,C-5-5, C17-20, ed. by Y. Kohasi (1968)

14.43 R.M. Gray, A. Buzo, A.H. Gray, Y. Matsuyama: Distortion measures for speech process, IEEE Trans. Acoust. Speech Signal Process. **ASSP-28**(4), 367–376 (1980)

14.44 K.K. Paliwal, W.B. Kleijn: Quantization of LPC parameters. In: *Speech Coding and Synthesis*, ed. by W.B. Kleijn, K.K. Paliwal (Elsevier, Amsterdam 1995) pp. 433–466

14.45 W.R. Gardner, B.D. Rao: Noncausal all-pole modeling of voiced speech, IEEE Trans. Speech Audio Process. **5**(1), 1–10 (1997)

14.46 M. Nilsson, W.B. Kleijn: Shannon entropy estimation based on high-rate quantization theory, Proc. EUSIPCO (2004) pp. 1753–1756

14.47 M. Nilsson: *Entropy and Speech* (Royal Institute of Technology, Stockholm 2006), Ph.D. dissertation, KTH

14.48 C. Lamm: *Improved Spectral Estimation in Speech Coding* (Lund Institute of Technology (LTH), Lund 1998), Master's thesis

14.49 K.L.C. Chan: Split-dimension vector quantization of parcor coefficients for low bit rate speech coding, IEEE Trans. Speech Audio Process. **2**(3), 443–446 (1994)

14.50 A. Subramaniam, B.D. Rao: PDF optimized parametric vector quantization of speech line spectral freuencies, IEEE Speech Coding Workshop (Delavan 2000) pp. 87–89

14.51 P. Hedelin, J. Skoglund: Vector quantization based on Gaussian mixture models, IEEE Trans. Speech Audio Process. **8**(4), 385–401 (2000)

14.52 S. Srinivasan, J. Samuelsson, W.B. Kleijn: Speech enhancement using a-priori information with classified noise codebooks, Proc. EUSIPCO (2004) pp. 1461–1464

14.53 W.R. Gardner, B.D. Rao: Optimal distortion measures for the high rate vector quantization of LPC parameters, Proc. IEEE ICASSP (1995) pp. 752–755

14.54 M.Y. Kim, W.B. Kleijn: KLT-based adaptive classified vector quantization of the speech signal, IEEE Trans. Speech Audio Process. **12**(3), 277–289 (2004)

14.55 P. Kroon, E.F. Deprettere: A class of analysis-by-synthesis predictive coders for high quality speech coding at rates between 4.8 and 16 kbit/s, IEEE J. Sel. Area. Commun. **6**(2), 353–363 (1988)

14.56 J. Chen, A. Gersho: Real-time vector APC speech coding at 4–800 bps with adaptive postfiltering, Proc. IEEE ICASSP (1987) pp. 2185–2188

14.57 J. Johnston: Transform coding of audio signals using perceptual noise criteria, IEEE J. Sel. Area. Commun. **6**(2), 314–323 (1988)

14.58 H. Malvar: Enhancing the performance of subband audio coders for speech signals, Proc. IEEE Int. Symp. on Circ. Syst., Vol. 5 (1998) pp. 98–101

14.59 R. Veldhuis: Bit rates in audio source coding, IEEE J. Sel. Area. Commun. **10**(1), 86–96 (1992)

14.60 B.C.J. Moore: Masking in the human auditory system. In: *Collected papers on digital audio bit-rate reduction*, ed. by N. Gilchrist, C. Grewin (Audio Eng. Soc., New York 1996)

14.61 B.C.J. Moore: *An Introduction to the Psychology of Hearing* (Academic, London 1997)

14.62 E. Zwicker, H. Fastl: *Psychoacoustics* (Springer Verlag, Berlin, Heidelberg 1999)

14.63 T. Painter, A. Spanias: Perceptual coding of digital audio, Proc. IEEE **88**(4), 451–515 (2000)

14.64 J.H. Plasberg, W.B. Kleijn: The sensitivity matrix: Using advanced auditory models in speech and audio processing, IEEE Trans. Speech Audio Process. **15**, 310–319 (2007)

14.65 J.L. Hall: Auditory psychophysics for coding applications. In: *The Digital Signal Processing Handbook*, ed. by V.K. Madisetti, D. Williams (CRC, Boca Raton 1998) pp. 39.1–39.25

14.66 W. Jesteadt, S.P. Bacon, J.R. Lehman: Forward masking as a function of frequency, masker level and signal delay, J. Acoust. Soc. Am. **71**(4), 950–962 (1982)

14.67 D. Sinha, J.D. Johnston: Audio compression at low bit rates using a signal adaptive switched

filterbank, Proc. IEEE ICASSP, Vol. 2 (1996) pp. 1053–1056
14.68 T. Verma, T. Meng: A 6 kbps to 85 kbps scalable audio coder, Proc. IEEE ICASSP, Vol. 2 (2000) pp. II877–II880
14.69 A.S. Scheuble, Z. Xiong: Scalable audio coding using the nonuniform modulated complex lapped transform, Proc. IEEE ICASSP, Vol. 5 (2001) pp. 3257–3260
14.70 R. Heusdens, R. Vafin, W.B. Kleijn: Sinusoidal modeling using psychoacoustic-adaptive matching pursuits, IEEE Signal Proc. Lett. **9**(8), 262–265 (2002)
14.71 M.Y. Kim, W.B. Kleijn: Resolution-constrained quantization with JND based perceptual-distortion measures, IEEE Signal Proc. Lett. **13**(5), 304–307 (2006)
14.72 O. Ghitza: Auditory nerve representation as a basis for speech processing. In: *Advances in Speech Signal Processing* (Dekker, New York 1992) pp. 453–485
14.73 T. Dau, D. Püschel, A. Kohlrausch: A quantitative model of the effective signal processing in the auditory system. I. Model structure, J. Acoust. Soc. Am. **99**(6), 3615–3622 (1996)
14.74 T. Dau, B. Kollmeier, A. Kohlrausch: Modeling auditory processing of amplitude modulation. I. detection and masking with narrowband carriers, J. Acoust. Soc. Am. **102**(5), 2892–2905 (1997)
14.75 G. Kubin, W.B. Kleijn: On speech coding in a perceptual domain, Proc. IEEE ICASSP, Vol. I (1999) pp. 205–208
14.76 F. Baumgarte: *Ein psychophysiologisches Gehörmodell zur Nachbildung von Wahrnehmungsschwellen für die Audiocodierung* (Univ. Hannover, Hannover 2000), Ph.D. dissertation (in German)
14.77 S. van de Par, A. Kohlrausch, G. Charestan, R. Heusdens: A new psychoacoustical masking model for audio coding applications, Proc. IEEE ICASSP (2002) pp. 1805–1808
14.78 D. Sen, D. Irving, W. Holmes: Use of an auditory model to improve speech coders, Proc. IEEE ICASSP (1993) pp. II411–II414
14.79 J.H. Plasberg, D.Y. Zhao, W.B. Kleijn: The sensitivity matrix for a spectro-temporal auditory model, Proc. EUSIPCO (2004) pp. 1673–1676
14.80 X. Yang, K. Wang, S. Shamma: Auditory representation of acoustic signals, IEEE Trans. Inform. Theory **38**(2), 824–839 (1996)
14.81 T. Linder, R. Zamir, K. Zeger: High-resolution source coding for non-difference measures: the rate-distortion function, IEEE Trans. Inform. Theory **45**(2), 533–547 (1999)
14.82 I. Gerson, M. Jasiuk: Vector sum excited linear prediction (VSELP), Proc. IEEE ICASSP (1990) pp. 461–464

15. Voice over IP: Speech Transmission over Packet Networks

J. Skoglund, E. Kozica, J. Linden, R. Hagen, W. B. Kleijn

The emergence of packet networks for both data and voice traffic has introduced new challenges for speech transmission designs that differ significantly from those encountered and handled in traditional circuit-switched telephone networks, such as the public switched telephone network (PSTN). In this chapter, we present the many aspects that affect speech quality in a voice over IP (VoIP) conversation. We also present design techniques for coding systems that aim to overcome the deficiencies of the packet channel. By properly utilizing speech codecs tailored for packet networks, VoIP can in fact produce a quality higher than that possible with PSTN.

15.1 Voice Communication

Voice over internet protocol (IP), known as VoIP, represents a new voice communication paradigm that is rapidly establishing itself as an alternative to traditional telephony solutions. While VoIP generally leads to cost savings and facilitates improved services, its quality has not always been competitive. For over a century, voice communication systems have used virtually exclusively circuit-switched networks and this has led to a high level of maturity. The end-user has been accustomed to a telephone conversation that has *consistent* quality and low delay. Further, the user expects a signal that has a narrow-band character and, thus, accepts the limitations present in traditional solutions, limitations that VoIP systems lack.

A number of fundamental differences exist between traditional telephony systems and the emerging VoIP systems. These differences can severely affect voice quality if not handled properly. This chapter will discuss the major challenges specific to VoIP and show that, with proper design, the quality of a VoIP solution can be significantly better than that of the public switched telephone network (PSTN). We first provide a broad overview of the issues that affect end-to-end quality. We then present some general techniques for designing speech coders that are suited for the challenges imposed by VoIP. We emphasize multiple description coding, a powerful paradigm that has shown promising performance in practical systems, and also facilitates theoretical analysis.

15.1.1 Limitations of PSTN

Legacy telephony solutions are narrow-band. This property imposes severe limitations on the achievable quality. In fact, in traditional telephony applications, the speech bandwidth is restricted more than the inherent

limitations of narrow-band coding at an 8 kHz sampling rate. Typical telephony speech is band-limited to 300–3400 Hz. This bandwidth limitation explains why we are used to expect telephony speech to sound weak, unnatural, and lack crispness. The final connection to most households (the so-called local loop) is generally analog, by means of two-wire copper cables, while entirely digital connections are typically only found in enterprise environments. Due to poor connections or old wires, significant distortion may be generated in the analog part of the phone connection, a type of distortion that is entirely absent in VoIP implementations. Cordless phones also often generate significant analog distortion due to radio interference and other implementation issues.

15.1.2 The Promise of VoIP

It is clear that significant sources of quality degradation exist in the PSTN. VoIP can be used to avoid this distortion and, moreover, to remove the basic constraints imposed by the analog connection to the household.

As mentioned above, even without changing the sampling frequency, the bandwidth of the speech signal can be enhanced over telephony band speech. It is possible to extend the lower band down to about 50 Hz, which improves the base sound of the speech signal and has a major impact on the naturalness, presence, and comfort in a conversation. Extending the upper band to almost 4 kHz (a slight margin for sampling filter roll-off is necessary) improves the naturalness and *crispness* of the sound. All in all, a fuller, more-natural voice and higher intelligibility can be achieved just by extending the bandwidth within the limitations of narrow-band speech. This is the first step towards *face-to-face* communication quality offered by wide-band speech.

In addition to having an extended bandwidth, VoIP has fewer sources of analog distortion, resulting in the possibility to offer significantly better quality than PSTN within the constraint of an 8 kHz sampling rate. Even though this improvement is often clearly noticeable, far better quality can be achieved by taking the step to wide-band coding.

One of the great advantages of VoIP is that there is no need to settle for narrow-band speech. In principle, compact disc (CD) quality is a reasonable alternative, allowing for the best possible quality. However, a high sampling frequency results in a somewhat higher transmission bandwidth and, more importantly, imposes tough requirements on hardware components. The bandwidth of speech is around 10 kHz [15.1], implying a sampling frequency of 20 kHz for good quality. However, 16 kHz has been chosen in the industry as the best trade-off between bit rate and speech quality for wide-band speech coding.

By extending the upper band to 8 kHz, significant improvements in intelligibility and quality can be achieved. Most notably, fricative sounds such as [s] and [f], which are hard to distinguish in telephony band situations, sound natural in wide-band speech.

Many hardware factors in the design of VoIP devices affect speech quality as well. Obvious examples are microphones, speakers, and analog-to-digital converters. These issues are also faced in regular telephony, and as such are well understood. However, since the limited signal bandwidth imposed by the traditional network is the main factor affecting quality, most regular phones do not offer high-quality audio. Hence, this is another area of potential improvement over the current PSTN experience.

There are other important reasons why VoIP is rapidly replacing PSTN. These include cost and flexibility. VoIP extends the usage scenarios for voice communications. The convergence of voice, data, and other media presents a field of new possibilities. An example is web collaboration, which combines application sharing, voice, and video conferencing. Each of the components, transported over the same IP network, enhances the experience of the others.

15.2 Properties of the Network

15.2.1 Network Protocols

Internet communication is based on the internet protocol (IP) which is a network layer (layer 3) protocol according to the seven-layer open systems interconnection (OSI) model [15.2]. The physical and data link layers reside below the network layer. On top of the network layer protocol, a transport layer (OSI layer 4) protocol is deployed for the actual data transmission. Most internet applications are using the transmission control protocol (TCP) [15.3] as the transport protocol. TCP is very robust since it allows for retransmission

in the case that a packet has been lost or has not arrived within a specific time. However, there are obvious disadvantages of deploying this protocol for real-time, two-way communication. First and foremost, delays can become very long due to the retransmission process. Another major disadvantage of TCP is the increased traffic load due to transmission of acknowledgements and retransmitted packets. A better choice of transport layer protocol for real-time communication such as VoIP is the user datagram protocol (UDP) [15.4]. UDP does not implement any mechanism for retransmission of packets and is thus more efficient than TCP for real-time applications. On top of UDP, another Internet Engineering Task Force (IETF) protocol, the real-time transport protocol (RTP) [15.5], is typically deployed. This protocol includes all the necessary mechanisms to transport data generated by both standard codecs as well as proprietary codecs.

It should be mentioned that recently it has become common to transmit VoIP data over TCP to facilitate communication through firewalls that would normally not allow VoIP traffic. This is a good solution from a connectivity point of view, but introduces significant challenges for the VoIP software designer due to the disadvantages with deploying TCP for VoIP.

15.2.2 Network Characteristics

Three major factors associated with packet networks have a significant impact on perceived speech quality: delay, jitter, and packet loss. All three factors stem from the nature of a packet network, which provides no guarantee that a packet of speech data will arrive at the receiving end in time, or even that it will arrive at all. This contrasts with traditional telephony networks where data are rarely, or never, lost and the transmission delay is usually a fixed parameter that does not vary over time. These network effects are the most important factors distinguishing speech processing for VoIP from traditional solutions. If the VoIP device cannot address network degradation in a satisfactory manner, the quality can never be acceptable. Therefore, it is of utmost importance that the characteristics of the IP network are taken into account in the design and implementation of VoIP products as well as in the choice of components such as the speech codec. In the following sub-sections delay, jitter, and packet loss are discussed and methods to deal with these challenges are covered.

A fact often overlooked is that both sides of a call need to have robust solutions even if only one side is connected to a poor network. A typical example is a wireless device that has been properly designed to be able to cope with the challenges in terms of jitter and packet loss typical of a wireless (WiFi) network which is connecting through an enterprise PSTN gateway. Often the gateway has been designed and configured to handle network characteristics typical of a well-behaved wired local-area network (LAN) and not a challenging wireless LAN. The result can be that the quality is good in the wireless device but poor on the PSTN side. Therefore, it is crucial that all devices in a VoIP solution are designed to be robust against network degradation.

Delay

Many factors affect the perceived quality in two-way communication. An important parameter is the transmission delay between the two end-points. If the latency is high, it can severely affect the quality and ease of conversation. The two main effects caused by high latency are annoying talker overlap and echo, which both can cause significant reduction of the perceived conversation quality.

In traditional telephony, long delays are experienced only for satellite calls, other long-distance calls, and calls to mobile phones. This is not true for VoIP. The effects of excessive delay have often been overlooked in VoIP design, resulting in significant conversational quality degradation even in short-distance calls. Wireless VoIP, typically over a wireless LAN (WLAN), is becoming increasingly popular, but increases the challenges of delay management further.

The impact of latency on communication quality is not easily measured and varies significantly with the usage scenario. For example, long delays are not perceived as annoying in a cell-phone environment as for a regular wired phone because of the added value of mobility. The presence of echo also has a significant impact on our sensitivity to delay: the higher the latency, the lower the perceived quality. Hence, it is not possible to list a single number for how high latency is acceptable, but only some guidelines.

If the overall delay is more than about 40 ms, an echo is audible [15.6]. For lower delays, the echo is only perceived as an expected side-tone. For longer delays a well-designed echo canceler can remove the echo. For very long delays (greater than 200 ms), even if echo cancelation is used, it is hard to maintain a two-way conversation without talker overlap. This effect is often accentuated by shortcomings of the echo canceler design. If no echo is generated, a slightly higher delay is acceptable.

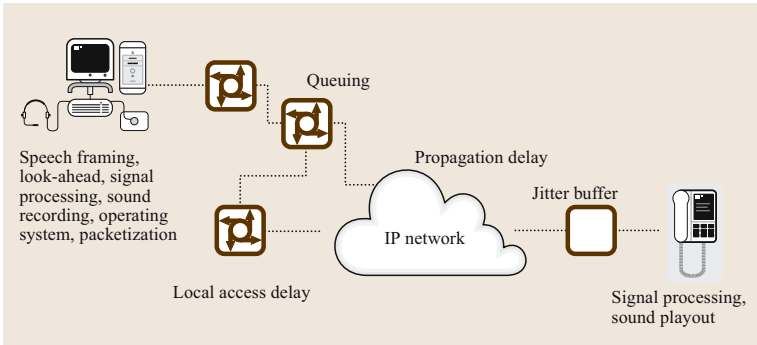


Fig. 15.1 Main delay sources in VoIP

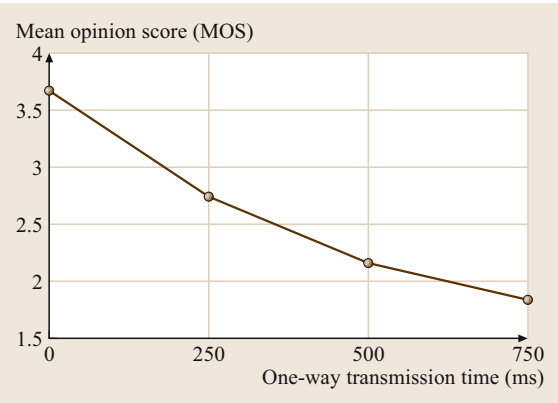


Fig. 15.2 Effect of delay on conversational quality from ITU-T G.114

The International Telecommunication Union – Telecommunication Standardization Sector (ITU-T) recommends in standard G.114 [15.7] that the one-way delay should be kept below 150 ms for acceptable conversation quality (Fig. 15.2 is from G.114 and shows the perceived effect on quality as a function of delay). Delays between 150 and 400 ms may be acceptable, but have an impact on the perceived quality of user applications. A latency larger than 400 ms is unacceptable.

Packet Loss

Packet losses often occur in the routers, either due to high router load or to high link load. In both cases, packets in the queues may be dropped. Packet loss also occurs when there is a breakdown in a transmission link. The result is data link layer error and the incomplete packet is dropped. Configuration errors and collisions may also result in packet loss. In non-real-time applications, packet loss is solved at the transfer layer by retransmission using TCP. For telephony, this is not a viable solution since transmitted packets would arrive too late for use.

When a packet loss occurs some mechanism for filling in the missing speech must be incorporated. Such solutions are usually referred to as packet loss concealment (PLC) algorithms (Sect. 15.5). For best performance, these algorithms have to accurately predict the speech signal and make a smooth transition between the previous decoded speech and inserted segment.

Since packet losses occur mainly when the network is heavily loaded, it is not uncommon for packet losses to appear in bursts. A burst may consist of a series of consecutive lost packets or a period of high packet loss rate. When several consecutive packets are lost, even good PLC algorithms have problems producing acceptable speech quality.

To save transmission bandwidth, multiple speech frames are sometimes carried in a single packet, so a single lost packet may result in multiple lost frames. Even if the packet losses occur more spread out, the listening experience is then similar to that of having the packet losses occur in bursts.

Network Jitter

The latency in a voice communication system can be attributed to algorithmic, processing, and transmission delays. All three delay contributions are constant in a conventional telephone network. In VoIP, the algorithmic and processing delays are constant, but the transmission delay varies over time. The transit time of a packet through an IP network varies due to queuing effects. The transmission delay is interpreted as consisting of two parts, one being the constant or slowly varying network delay and the other being the rapid variations on top of the basic network delay, usually referred to as jitter.

The jitter present in packet networks complicates the decoding process in the receiver device because the

decoder needs to have packets of data available at the right time instants. If the data is not available, the decoder cannot produce continuous speech. A jitter buffer is normally used to make sure that packets are available when needed.

Clock Drift

Whether the communication end-points are gateways or other devices, low-frequency clock drift between the two can cause receiver buffer overflow or underflow. Simply speaking, this effect can be described as the two devices talking to each other having different time references. For example, the transmitter might send packets every 20 ms according to its perception of time, while the receiver's perception is that the packets arrive every 20.5 ms. In this case, for every 40th packet, the receiver has to perform a packet loss concealment to avoid buffer underflow. If the clock drift is not detected accurately, delay builds up during a call, so clock drift can have a significant impact on the speech quality. This is particularly difficult to mitigate in VoIP.

The traditional approach to address clock drift is to deploy a clock synchronization mechanism at the receiver to correct for clock drift by comparing the time stamps of the received RTP packets with the local clock. It is hard to obtain reliable clock drift estimates in VoIP because the estimates are based on averaging packet arrivals at a rate of typically 30–50 per second and because of the jitter in their arrival times. Consider for comparison the averaging on a per-sample basis at a rate of 8000 per second that is done in time-division multiplexing (TDM) networks [15.8]. In practice many algorithms designed to mitigate the clock drift effect fail to perform adequately.

Wireless Networks

Traditionally, packet networks consisted of wired Ethernet solutions that are relatively straightforward to manage. However, the rapid growth of wireless LAN (WLAN) solutions is quickly changing the network landscape. WLAN, in particular the IEEE 802.11 family of standards [15.9], offers mobility for computer access and also the flexibility of wireless IP phones, and are hence of great interest for VoIP systems. Jitter and effective packet loss rates are significantly higher in WLAN than in a wired network, as mentioned in Sect. 15.2.3. Furthermore, the network characteristics often change rapidly over time. In addition, as the user moves physically, coverage and interference from other radio sources–such as cordless phones, Bluetooth [15.10] devices, and microwave ovens-varies. The result is that high-level voice quality is significantly harder to guarantee in a wireless network than a typical wired LAN.

WLANs are advertised as having very high throughput capacity (11 Mb/s for 802.11b and 54 Mb/s for 802.11a and 802.11g). However, field studies show that actual throughput is often only half of this, even when the client is close to the access point. It has been shown that these numbers are even worse for VoIP due to the high packet rate, with typical throughput values of 5–10% (Sect. 15.2.3).

When several users are connected to the same wireless access point, congestion is common. The result is jitter that can be significant, particularly if large data packets are sent over the same network. The efficiency of the system quickly deteriorates when the number of users increases.

When roaming in a wireless network, the mobile device has to switch between access points. In a traditional WLAN, it is common that such a hand-off introduces a 500 ms transmission gap, which has a clearly audible impact on the call quality. However, solutions are now available that cut that delay number to about 20 to 50 ms, if the user is not switching between two IP subnets. In the case of subnet roaming the handover is more complicated and no really good solutions exist currently. Therefore, it is common to plan the network in such a way that likelihood of subnet roaming is minimized.

Sensitivity to congestion is only one of the limitations of 802.11 networks. Degraded link quality, and consequently reduced available bandwidth, occurs due to a number of reasons. Some 802.11 systems operate in the unlicensed 2.4 GHz frequency range and share this spectrum with other wireless technologies, such as Bluetooth and cordless phones. This causes interference with potentially severe performance degradation since a lower connection speed than the maximum is chosen.

Poor link quality also leads to an increased number of retransmissions, which directly affects the delay and jitter. The link quality varies rapidly when moving around in a coverage area. This is a severe drawback, since a WLAN is introduced to add mobility and a wireless VoIP user can be expected to move around the coverage area. Hence, the introduction of VoIP into a WLAN environment puts higher requirements on network planning than for an all-data WLAN.

The result of the high delays that occur due to access-point congestion and bad link quality is that the packets often arrive too late to be useful. Therefore, the effective packet loss rate after the jitter buffer is typically significantly higher for WLANs than for wired LANs.

15.2.3 Typical Network Characteristics

Particularly VoIP communications that involve many *hops* are often negatively affected by delay, packet loss, and jitter. Little published work is available that describes network performance quantitatively. In one study towards improved understanding of network behavior, the transmission characteristics of the internet between several end-points were monitored over a four-week period in early 2002. Both packet loss and jitter were first measured as an average over 10 s periods. The maximum of these values over 5 min were then averaged over 7 d. Table 15.1 summarizes the results of these measurements. The table shows that significant jitter is present in long-distance IP communication, which affects speech quality significantly. Also, it was noted that, compared to Europe and the US, degradation in the quality of communications was more severe with calls to, from, and within Asia.

The ratio of infrastructure to traffic demand determines the level of resource contention. This, in turn, affects packet loss, delay, and jitter. With larger networks, packets can avoid bottlenecks and generally arrive within a reasonable time. When network conditions are less than ideal, communication quality deteriorates rapidly.

An informal test of the capacity of a wireless network was presented in [15.11]. The impact on packet loss and jitter of the number of simultaneous calls over a wireless access point with perfect coverage for all users is depicted in Figs. 15.3 and 15.4. Each call used the ITU-T G.711 codec [15.12] with a packet size of 20 ms which, including IP headers, results in a payload bandwidth of 80 kb/s. These results show that, for this access point, only five calls can be allowed if we do not allow packet loss. The results for this access point correspond to a bandwidth utilization factor of less than 10% percent. Interestingly, using a higher-compression speech codec did not increase the number of channels that can be handled. The reason is that access-point congestion depends much more on the number of packets the access point has to process than on the sum of the user bit rates. Voice packets are small and sent very frequently, which explains the low throughput for voice packets. Because of this limitation it is common to put several voice frames into the same packet, which reduces the number of packets and hence increases the throughput. However, this results both in

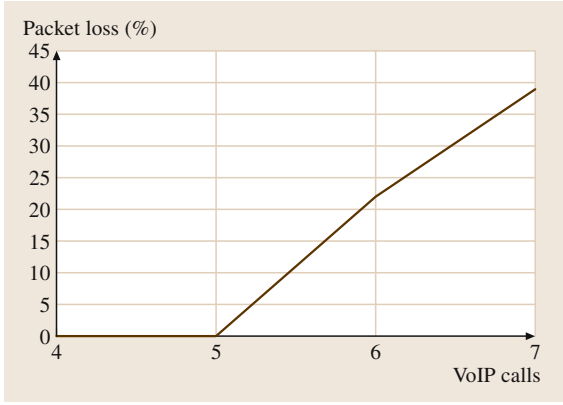


Fig. 15.3 Effect of access point congestion on the amount of packet loss as a function of the number of simultaneous VoIP calls through one access point (after [15.11])

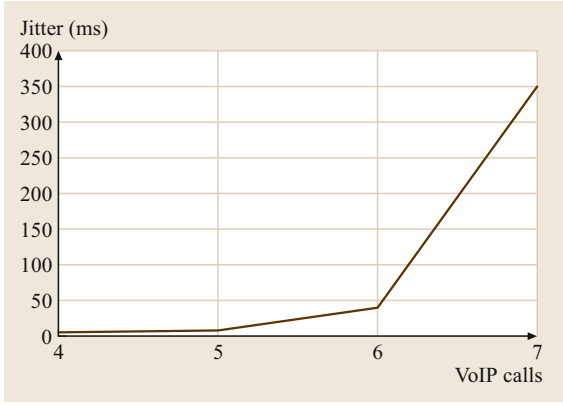


Fig. 15.4 Effect of access point congestion on network jitter as a function of the number of simultaneous VoIP calls through one access point (after [15.11])

Table 15.1 Results of international call performance monitoring. Long-term averages of short-term maximum values

Connection	Packet loss (%)	Roundtrip delay (ms)	Jitter (ms)	Hops
Hong Kong – Urumgi, Xinjiang	50	800	350	20
Hong Kong – San Francisco	25	240	250	17
San Francisco – Stockholm	8	190	200	16–18

increased delay and in an increased impact of packet loss.

15.2.4 Quality-of-Service Techniques

Since network imperfections have a direct impact on the voice quality, it is important that the network is properly designed and managed. Measures taken to ensure network performance are usually referred to as quality-of-service (QoS) techniques. High quality of service can be achieved by adjusting capacity, by packet prioritization, and by network monitoring.

Capacity Management

Generally capacity issues are related to the connection to a wide-area network or other access links. It is uncommon that a local-area network has significant problems in this respect.

Prioritization Techniques

By prioritizing voice over less time-critical data in routers and switches, delay and jitter can be reduced significantly without a noticeable effect on the data traffic. Many different standards are available for implementing differentiated services, including IEEE 802.1p/D [15.13] and DiffServ [15.14]. In addition, the resource reservation protocol (RSVP) [15.15] can be used to reserve end-to-end bandwidth for the voice connection.

Network Monitoring

As the needs and requirements of networks continuously change, it is important to implement ongoing monitoring and management of the network.

Clearly, QoS management is easily ensured in an isolated network. Challenges typically arise in cases where the traffic is routed through an unmanaged network. A particularly interesting and challenging scenario is a telecommuter connecting to the enterprise network through a virtual private network. For more information on QoS, we refer the reader to the vast literature available on this topic, e.g. [15.16].

The QoS supplement developed in the Institute of Electrical and Electronics Engineers (IEEE) is called 802.11e [15.17] and is applicable to 802.11a, 802.11b and 802.11g. The development of this standard has been quite slow, and it likely will take time before significant deployment is seen. In the meanwhile, several vendors have developed proprietary enhancements of the standards that are already deployed. The introduction of some QoS mechanisms in WLAN environments will have a significant impact on the number of channels that can be supported and the amount of jitter present in the system. However, it is clear that the VoIP conditions in WLANs will remain more challenging than those of the typical wired LAN. Hence, particularly in the case of WLAN environments, QoS has to be combined with efficient jitter buffer implementations and careful latency management in the system design.

15.3 Outline of a VoIP System

VoIP is used to provide telephony functionality to the end user, which can communicate through various devices. For traditional telephony replacement, regular phones are used while so-called media gateways to the IP network convert the calls to and from IP traffic. These gateways can be large trunking devices, situated in a telephony carrier's network, or smaller gateways, e.g., one-port gateways in an end user's home. An IP phone, on the other hand, is a device that looks very much like a regular phone, but it connects directly to an

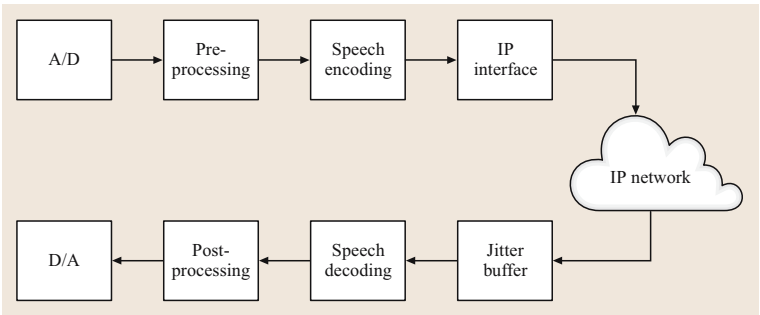


Fig. 15.5 A typical VoIP system

IP network. Lately, PCs have become popular devices for VoIP through services like Google Talk, Skype, Yahoo! Messenger and others. In this case, the phone is replaced by an application running on the PC that provides the telephony functionality. Such applications also exist for WiFi personal digital assistants (PDAs) and even for cell phones that have IP network connections. Hence, it is not an overstatement to say that the devices used in VoIP represent a changing environment and impose challenges for the speech processing components. Therefore, the systems deployed need to be able to cope with the IP network as described in the previous section, as well as the characteristics of the different applications.

A simplified block diagram of the speech processing blocks of a typical VoIP system is depicted in Fig. 15.5. At the transmitting end, the speech signal is first digitized in an analog-to-digital (A/D) converter. Next, a preprocessing block performs functions such as echo cancelation, noise suppression, automatic gain control, and voice activity detection, depending on the needs of the system and the end user's environment. Thereafter, the speech signal is encoded by a speech codec and the resulting bit stream is transmitted in IP packets. After the packets have passed the IP network, the receiving end converts the packet stream to a voice signal using the following basic processing blocks: a jitter buffer receiving the packets, speech decoding and postprocessing, e.g., packet loss concealment.

Next, we look closer at the major speech processing components in a VoIP system. These are echo cancelation, speech coding, jitter buffering and packet loss recovery. Further, we provide an overview of a number of auxiliary speech processing components.

15.3.1 Echo Cancelation

One of the most important aspects in terms of effect on the end-to-end quality is the amount of echo present during a conversation. An end user experiences echo by hearing a delayed version of what he or she said played back. The artifact is at best annoying and sometimes even renders the communication useless. An echo cancelation algorithm that estimates and removes the echo is needed. The requirements on an echo canceler to achieve good voice quality are very challenging. The result of a poor design can show up in several ways, the most common being:

1. audible residual echo, due to imperfect echo cancelation,
2. clipping of the speech, where parts of or entire words disappear due to too much cancelation,
3. poor double-talk performance.

The latter problem occurs when both parties attempt to talk at the same time and the echo canceler suppresses one or both of them leading to unnatural conversation. A common trade-off for most algorithms is the performance in double-talk versus single-talk, e.g., a method can be very good at suppressing echo, but has clipping and double-talk artifacts or vice versa [15.18].

Echo is a severe distraction if the round trip delay is longer than 30–40 ms [15.6]. Since the delays in IP telephony systems are significantly higher, the echo is clearly audible to the speaker. Canceling echo is, therefore, essential to maintaining high quality. Two types of echo can deteriorate speech quality: network echo and acoustic echo.

Network Echo Cancelation

Network echo is the dominant source of echo in telephone networks. It results from the impedance mismatch at the hybrid circuits of a PSTN exchange, at which the two-wire subscriber loop lines are connected to the four-wire lines of the long-haul trunk. The echo path is stationary, except when the call is transferred to another handset or when a third party is connected to the phone call. In these cases, there is an abrupt change in the echo path. Network echo in a VoIP system occurs through gateways, where there is echo between the incoming and outgoing signal paths of a channel generated by the PSTN network. Network echo cancelers for VoIP are implemented in the gateways.

The common solutions to echo cancelation and other impairments in packet-switched networks are basically adaptations of techniques used for the circuit-switched network. To achieve the best possible quality, a systematic approach is necessary to address the quality-of-sound issues that are specific to packet networks. Therefore, there may be significant differences between the "repackaged" circuit-switched echo cancelers and echo cancelers optimized for packet networks.

Acoustic Echo Cancelation

Acoustic echo occurs when there is a feedback path between a telephone's microphone and speaker (a problem primarily associated with wireless and hands-free devices) or between the microphone and the speakers of a personal computer (PC)-based system. In addition to the direct coupling path from microphone to speaker, acoustic echo can be caused by multiple reflections of

the speaker's sound waves back to the microphone from walls, floor, ceiling, windows, furniture, a car's dashboard, and other objects. Hence, the acoustic echo path is nonstationary. Acoustic echo has become a major issue in VoIP systems as more direct end-to-end VoIP calls are being made, e.g., by the very popular PC-based services.

There are few differences between designing acoustic echo cancelation (AEC) algorithms for VoIP and for traditional telephony applications. However, due to the higher delays typically experienced in VoIP, the requirements on the AEC are often more demanding. Also, wide-band speech adds some new challenges in terms of quality and complexity. Large-scale deployments in PC environments create demanding challenges in terms of robustness to different environments (e.g., various microphones and speakers) as well as varying echo paths created by non real-time operating systems, like Windows.

15.3.2 Speech Codec

The basic algorithmic building block in a VoIP system, that is always needed, is the speech codec. When initiating a voice call, a session protocol is used for the call setup, where both sides agree on which codec to use. The endpoints normally have a list of available codecs to choose from. The most common codec used in VoIP is the ITU-T G.711 [15.12] standard codec, which is mandatory for every end point to support for interoperability reasons, i. e., to guarantee that one common codec always exist so that a call can be established. For low-bandwidth applications, ITU-T G.729 [15.19] has been the dominant codec.

The quality of speech produced by the speech codec defines the upper limit for achievable end-to-end quality. This determines the sound quality for perfect network conditions, in which there are no packet losses, delays, jitter, echoes or other quality-degrading factors. The bit rate delivered by the speech encoder determines the bandwidth load on the network. The packet headers (IP, UDP, RTP) also add a significant portion to the bandwidth. For 20 ms packets, these headers increase the bit rate by 16 kbits/s, while for 10 ms packets the overhead bit rate doubles to 32 kbit/s. The packet header overhead versus payload trade-off has resulted in 20 ms packets being the most common choice.

A speech codec used in a VoIP environment must be able to handle lost packets. Robustness to lost packets determines the sound quality in situations where network congestion is present and packet losses are likely. Traditional circuit switched codecs, e.g., G.729, are vulnerable to packet loss due to inter-frame dependencies caused by the encoding method. A new codec without interframe dependencies called the internet low-bit-rate codec (iLBC) has been standardized by the IETF for VoIP use [15.20]. Avoiding interframe dependencies is one step towards more robust speech coding. However, even codecs with low interframe dependencies need to handle inevitable packet losses and we will discuss that in more detail later.

Increasing the sampling frequency from 8 kHz, which is used for narrow-band products, to 16 kHz, which is used for wide-band speech coding, produces more natural, comfortable and intelligible speech. However, wide-band speech coding has thus far found limited use in applications such as video conferencing because speech coders mostly interact with the PSTN, which is inherently narrow-band. There is no such limitation when the call is initiated and terminated within the IP network. Therefore, because of the dramatic quality improvement attainable, the next generation of speech codecs for VoIP will be wide-band. This is seen in popular PC clients that have better telephony quality.

If the call has to traverse different types of networks, the speech sometimes needs to be transcoded. For example, if a user on an IP network that uses G.729 is calling a user on the PSTN, the speech packets need to be decoded and reencoded with G.711 at the media gateway. Transcoding should be avoided if possible, but is probably inevitable until there is a unified all-IP network.

It lies in the nature of a packet network that it, over short periods of time, has a variable-throughput bandwidth. Hence, speech coders that can handle variable bit rate are highly suitable for this type of channels. Variable bit rate can either be source-controlled, network-controlled, or both. If the encoder can get feedback from the network about current packet loss rates, it can adapt its rate to the available bandwidth. An example of an adaptive-rate codec is described in [15.21].

15.3.3 Jitter Buffer

A jitter buffer is required at the receiving end of a VoIP call. The buffer removes the jitter in the arrival times of the packets, so that there is data available for speech decoding when needed. The exception is if a packet is lost or delayed more than the length the jitter buffer is set to handle. The cost of a jitter buffer is an increase in the overall delay. The objective of a jitter buffer algorithm

is to keep the buffering delay as short as possible while minimizing the number of packets that arrive too late to be used. A large jitter buffer causes an increase in the delay and decreases the packet loss. A small jitter buffer decreases the delay but increases the resulting packet loss. The traditional approach is to store the incoming packets in a buffer (packet buffer) before sending them to the decoder. Because packets can arrive out of order, the jitter buffer is not a strict first-in first-out (FIFO) buffer, but it also reorders packets if necessary.

The most straightforward approach is to have a buffer of a fixed size that can handle a given fixed amount of jitter. This results in a constant buffer delay and requires no computations and provides minimum complexity. The drawback with this approach is that the length of the buffer has to be made sufficiently large that even the worst case can be accommodated or the quality will suffer.

To keep the delay as short as possible, it is important that the jitter buffer algorithm adapts rapidly to changing network conditions. Therefore, jitter buffers with dynamic size allocation, so-called adaptive jitter buffers, are now the most common [15.22]. The adaptation is achieved by inserting packets in the buffer when the delay needs to be increased and removing packets when the delay can be decreased. Packet insertion is usually done by repeating the previous packet. Unfortunately, this generally results in audible distortion. To avoid quality degradation, most adaptive jitter buffer algorithms are conservative when it comes to reducing the delay to lessen the chance of further delay increases. The traditional packet buffer approach is limited in its adaptation granularity by the packet size, since it can only change the buffer length by adding or discarding one or several packets. Another major limitation of traditional jitter buffers is that, to limit the audible distortion of removing packets, they typically only function during periods of silence. Hence, delay builds up during a talk spurt, and it can take several seconds before a reduction in the delay can be achieved.

15.3.4 Packet Loss Recovery

The available methods to recover from lost packets can be divided into two classes: sender-and-receiver-based and receiver-only-based techniques. In applications where delay is not a crucial factor automatic repeat request (ARQ) is a powerful and commonly used sender-and-receiver-based technique. However, the delay constraint restricts the use of ARQ in VoIP and other methods to recover the missing packets must be considered. Robust encoding refers to methods where redundant side-information is added to the transmitted data packets. In Sect. 15.4, we describe robust encoding in more detail. Receiver-only-based methods, often called packet loss concealment (PLC), utilize only the information in previously received packets to replace the missing packets. This can be done by simply inserting zeros, repeating signals, or by some more-sophisticated methods utilizing features of the speech signal (e.g., pitch periods). Section 15.5 provides an overview of different error concealment methods.

15.3.5 Joint Design of Jitter Buffer and Packet Loss Concealment

It is possible to combine an advanced adaptive jitter buffer control with packet loss concealment into one unit [15.23]. The speech decoder is used as a *slave* in the sense that it decodes data and delivers speech segments back when the control logic asks for it. The new architecture makes the algorithm capable of adapting the buffer size on a millisecond basis. The approach allows it to quickly adapt to changing network conditions and to ensure high speech quality with minimal buffer latency. This can be achieved because the algorithm is working together with the decoder and not in the packet buffer. In addition to minimizing jitter buffer delay, the packet loss concealment part of the algorithm in [15.23] is based on a novel approach, and is capable of producing higher quality than any of the standard PLC methods. Experiments show that, with this type of approach, one-way delay savings of 30–80 ms are achievable in a typical VoIP environment [15.23]. Similar approaches have also been presented in, e.g., [15.24] and [15.25].

15.3.6 Auxiliary Speech Processing Components

In addition to the most visible and well-known voice processing components in a VoIP device there are many other important components that are included either to enhance the user experience or for other reasons, such as, reducing bandwidth requirements. The exploitation of such components depends on system requirements and usage scenarios, e.g., noise suppression for hands-free usage in noisy environment or voice activity detection to reduce bandwidth on bandlimited links.

Since no chain is stronger than its weakest link, it is imperative that even these components are designed in an appropriate fashion. Examples of such components include automatic gain control (AGC), voice activity de-

tection (VAD), comfort noise generation (CNG), noise suppression, and signal mixing for multiparty calling features. Typically, there is not much difference in the design or requirements between traditional telecommunications solutions and VoIP solutions for this type of components. However, for example VAD and CNG are typically deployed more frequently in VoIP systems. The main reason for the increased usage of VAD is that the net saving in bandwidth is very significant, due to the protocol overhead in VoIP. Also, an IP network is well suited to utilize the resulting variable bit rate to transport other data while no voice packets are sent.

VAD is used to determine silent segments, where packets do not need to be transmitted or, alternatively, only a sparse noise description is transmitted. The CNG unit produces a natural sound noise signal at the receiver, based either on a noise description that it receives or on an estimate of the noise. Due to misclassifications in the VAD algorithm clipping of the speech signal and noise bursts can occur. Also, since only comfort noise is played out during silence periods, the background signal may sound artificial. The most common problem with CNG, is that the signal level is set too low, which results in the feeling that the other person has dropped out of the conversation. These performance issues mandate that VAD should be used with caution to avoid unnecessary quality degradation.

Implementing multiparty calling also faces some challenges due to the characteristics of the IP networks. For example, the requirements for delay, clock drift, and echo cancelation performance are tougher due to the fact that several signals are mixed together and that there are several listeners present. A jitter buffer with low delay and the capability of efficiently handling clock drift offers a significant improvement in such a scenario. Serious complexity issues arise since different codecs can be used for each of the parties in a call. Those codecs might use different sampling frequencies. Intelligent schemes to manage complexity are thus important. One way to reduce the complexity is by using VAD to determine what participants are active at each point in time and only mix those into the output signals.

15.3.7 Measuring the Quality of a VoIP System

Speech quality testing methods can be divided into two groups: subjective and objective. Since the ultimate goal of speech quality testing is to get a measure of the perceived quality by humans, subjective testing is the more relevant technique. The results of subjective tests are often presented as mean opinion scores (MOS) [15.26]. However, subjective tests are costly and quality ratings are relative from user to user. As a result, automatic or *objective* measuring techniques were developed to overcome these perceived shortcomings. The objective methods provide a lower cost and simple alternative, however they are highly sensitive to processing effects and other impairments [15.27, 28].

Perceptual evaluation of speech quality (PESQ), defined in ITU-T recommendation P.862 [15.29], is the most popular objective speech quality measurement tool. Even though PESQ is recognized as the most accurate objective method the likelihood that it will differ more than 0.5 MOS from subjective testing is 30% [15.30]. It is obvious that the objective methods do not offer the necessary level of accuracy in predicting the perceived speech quality and that subjective methods have to be used to achieve acceptable accuracy.

15.4 Robust Encoding

The performance of a speech coder over a packet loss channel can efficiently be improved by adding redundancy at the encoding side and utilizing the redundancy at the decoder to fully or partly recover lost packets. The amount of redundancy must obviously be a function of the amount of packet loss. In this section we distinguish two classes of such so-called robust encoding approaches, *forward error correction* (FEC) and *multiple description coding* (MDC), and describe them in more detail. MDC is the more powerful of the two in the sense that it is more likely to provide graceful degradation.

15.4.1 Forward Error Correction

In FEC, information in lost packets can be recovered by sending redundant data together with the information. Here we distinguish FEC methods from other methods that introduce redundancy in that the additional data does not contain information that can yield a better reconstruction at the decoder if no packets were lost. The typical characteristics of FEC are that the performance with some packet loss is the same as with no packet loss but that the performance rapidly deteriorates (a *cliff* effect) at losses higher than a certain critical packet loss

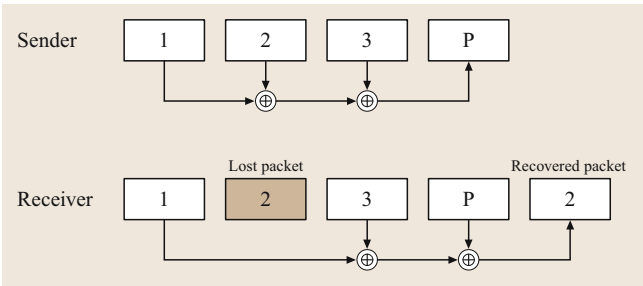


Fig. 15.6 FEC by parity coding. Every n-th transmitted packet is a parity packet constructed by bitwise XOR on the previous $n-1$ packets

rate determined by the amount of redundancy. There are two classes of FEC schemes: *media-independent FEC* and *media-dependent FEC*.

Media-Independent FEC

In media-independent FEC, methods known from channel coding theory are used to add blocks of parity bits. These methods are also known as erasure codes. The simplest of these, called parity codes, utilize exclusive-or (XOR) operations between packets to generate parity packets. One simple example is illustrated in Fig. 15.6, where every n-th packet contains bitwise XOR on the $n-1$ previous packets [15.31]. This method can exactly recover one lost packet if packet losses are at least separated by n packets. More-elaborate schemes can be achieved by different combinations of packets for the XOR operation. By increasing the amount of redundancy and delay it is possible to correct a small set of consecutively lost packets [15.32]. More-powerful error correction can be achieved using erasure codes such as

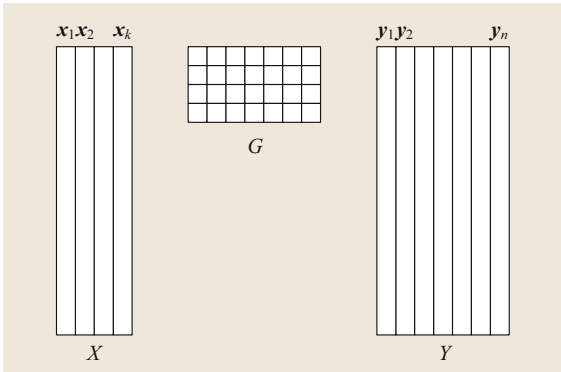


Fig. 15.7 FEC by an RS(n,k) code-sending side. A sequence of k data packets is multiplied by a generator matrix to form n encoded packets

Reed–Solomon (RS) codes. These codes were originally presented for streams of individual symbols, but can be utilized for blocks (packets) of symbols [15.33]. The channel model is in this case an erasure channel where packets are either fully received or erased (lost). From k packets of data an RS(n,k) code produces n packets of data such that the original k packets can be exactly recovered by receiving any subset of k (or more) packets. Assume each packet $\boldsymbol{x} = [x_1, x_2, \ldots, x_B]^{\mathrm{T}}$ contains B r-bit symbols $x_i = (b_i^{(1)}, b_i^{(2)}, \ldots, b_i^{(r)})$, $b_i^{(j)} \in [0,1]$.

The n packets can then be generated by (Fig. 15.7)

$$\mathrm{Y} = \mathrm{XG}\,, \tag{15.1}$$

where $\mathrm{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k]$, $\mathrm{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n]$, and G is a generator matrix. All operations are performed in the finite extension field $GF(2^r)$.The generator matrix is specific for the particular RS code. It is often convenient to arrange the generator matrix in systematic form, which yields an output where the first k packets are identical to the k uncoded packets and the other $n-k$ packets contain parity symbols exclusively. An example of constructing a generator matrix in systematic form is [15.33]

$$\mathrm{G} = \mathrm{V}_{k,k}^{-1}\mathrm{V}_{k,n}\,, \tag{15.2}$$

using the Vandermonde matrix

$$\mathrm{V}_{i,j} = \alpha^{ij}\,, \tag{15.3}$$

where α is a generating element in $GF(2^r)$. The r-bit symbols are all elements in this field.

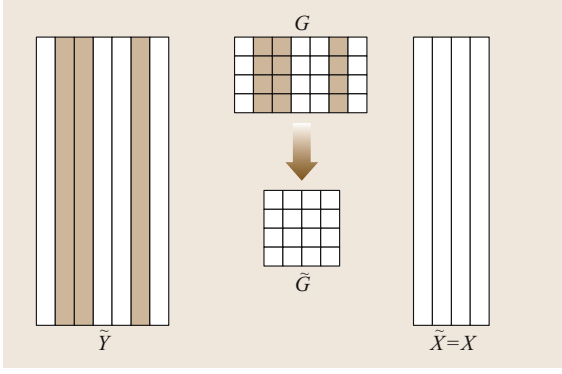


Fig. 15.8 FEC by an RS(n,k) code-receiving side. Some packets are lost during transmission (indicated by the gray color). The decoder forms a matrix $\tilde{G}$ from the columns corresponding to k correctly received packets. These packets are then multiplied with the inverse of $\tilde{G}$ to recover X, the original sequence of k data packets

From k available packets the receiver forms $\tilde{Y}$, a subset of Y. The recovered packets are then calculated as

$$\tilde{X} = \tilde{Y}\tilde{G}^{-1} , \tag{15.4}$$

where $\tilde{G}$ is formed by the k columns of G corresponding to the received packets. Figure 15.8 illustrates the decoding procedure. RS codes belong to a class of codes called maximum distance separable (MDS) codes that are generally powerful for many types of erasure channels but require a large n. For real-time interactive audio, delay is crucial and only short codes are feasible. Typical examples of RS codes utilized in VoIP are RS(5, 3) [15.34] and RS(3, 2) [15.35]. For the application of bursty erasure channels another type of codes, *maximally short codes* [15.36], have shown to require lower decoding delay than MDS codes.

A common way to decrease the packet overhead in FEC is to attach the redundant packets onto the information packet, a technique called *piggybacking*. Figure 15.9 depicts a case for an RS(5, 3) code, where the two parity packets are piggybacked onto the first two data packets of the next packet sequence.

Media-Dependent FEC

In media-dependent FEC, the source signal is encoded more than once. A simple scheme is just to encode the information twice and send the data in two separate packets. The extra packet can be piggybacked onto the subsequent packet. To lower the overall bit rate, it is more common that the second description uses a lower rate-compression method, resulting in a lower quality in case a packet needs to be recovered. The latter method is sometimes referred to as low-bit-rate redundancy (LBR) and has been standardized by IETF [15.37], which actually also provides procedures for the first method. In the LBR context, the original (high-quality) coded description is referred to as the *primary encoding* and the lower-quality description is denoted the *redundant encoding*. Examples of primary and redundant encodings are G.711 (64 kbps)+ GSM (13 kbps) [15.38] and G.729 (8 kbps)+ LPC10 (2.4 kbps) [15.35]. Figure 15.10 depicts the method for the case when the secondary encoding is piggybacked on the next primary encoding in the following packet. To safeguard against burst errors, the secondary description can be attached to the m-th next packet instead of the immediate subsequent packet, at the cost of a decoding delay of m packets. Even better protection is obtained with more redundancy, e.g., packet n contains the primary encoding of block data block n and redundant encodings of blocks $n-1, n-2, \ldots, n-m$ [15.38]. Although media-dependent FEC seems more popular, erasure codes are reported to have better subjective performance at comparable bit rates [15.35].

It is important to point out that most practical systems that today use FEC simply put the same encoded frame in two packets, as mentioned above. The rea-

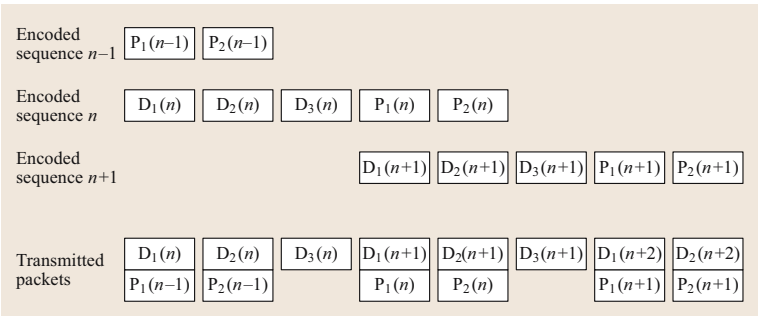


Fig. 15.9 Piggybacking in an RS(5, 3) FEC system. The parity packets of sequence n are attached to the data packets of sequence $n+1$.

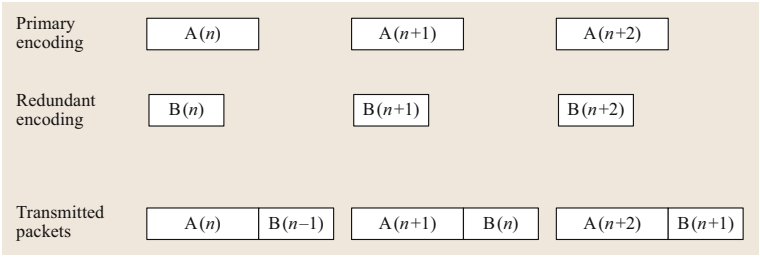


Fig. 15.10 Media-dependent FEC proposed by IETF. The data is encoded twice, and the additional redundant encoding is piggybacked onto the primary encoding in the next packet

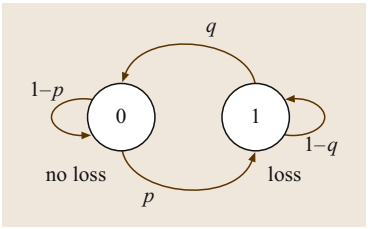


Fig. 15.11 The Gilbert two-state model of a bursty packet loss channel

son for doing so is due to the fact that a lower bit rate secondary coder is typically more complex than the primary encoder and the increase in bit rate is preferred to an increase in complexity.

Adaptive FEC

The amount of redundancy needed is a function of the degree of packet loss. Some attempts have been made to model the loss process in order to assess the benefits of adding redundancy. In [15.38] *Bolot* et al. used LBR according to the IETF scheme mentioned in the last paragraph and a Gilbert channel model (a Markov chain model for the dependencies between consecutive packet losses, Fig. 15.11) to obtain expressions for the perceived quality at the decoder as a function of packet loss and redundancy. In the derivations the perceived quality is assumed to be an increasing function of the bit rate. They propose a design algorithm to allocate redundancy that functions even in the case that there are only a handful of low bit rate secondary encodings to choose from. In the case where the channel packet loss can be estimated somewhere in the network and signalled back to the sender, e.g., through RTCP (an out-of-band signaling for RTP [15.5]), their algorithm can be utilized for adaptive FEC. Due to the adaptation it is possible to obtain a more graceful degradation of performance than the typical collapsing behavior of FEC.

15.4.2 Multiple Description Coding

Multiple description coding (MDC) is another method to introduce redundancy into the transmitted description to counter packet loss. Compared to erasure codes, the method has the advantage that it naturally leads to graceful degradation. It optimizes an average distortion criterion assuming a particular packet loss scenario, which can in principle consist of an ensemble of scenarios. Disadvantages of the multiple description coding technique are that it is difficult to combine multiple description coding with legacy coding systems and that changing the robustness level implies changing the entire source coder.

MDC Problem Formulation

Consider a sampled speech signal that is transmitted with packets. To facilitate the derivations that follow below, the speech signal contained in a packet is approximated as a weakly stationary and ergodic process that is independent and identically distributed (i.i.d.). We denote this speech process by X (thus, X denotes a sequence of random variables that are i.i.d.). We note that a process X with the aforementioned properties can be obtained from speech by performing an appropriate invertible transform. Optimal encoding of X over a channel facilitating a given rate R requires the *one* description of the source, which, when decoded at the receiver end, achieves the lowest achievable distortion $E[d(X, \hat{X})]$, where $\hat{X}$ is the reconstructed process. Varying the rate leads to a lower bound on all achievable distortions as a function of the rate, i. e., the distortion-rate function $D(R)$.

An extension of the problem formulation, to the case where several channels are available, is not entirely intuitive. We consider here only the case of two channels, $s \in \{1, 2\}$, each with rate R_s and corresponding description I_s. The channels are such that they either do or do not work. (This corresponds to the cases where the packet either arrives or does not arrive.) Thus, we can distinguish four receiver scenarios: we receive both descriptions, we receive description I_1 solely, we receive description I_2 solely, or we receive no description. A particular formulation of the objective of multiple description coding is to find the two descriptions that minimize the central distortion $E[d(X, \hat{X}_0)]$, with constraints on the side distortions, $E[d(X, \hat{X}_s)] < D_s$, where $\hat{X}_0$ and $\hat{X}_s$ are *central* and *side* reconstruction values obtained from both descriptions and a single description, respectively.

Figure 15.12 illustrates the operation of multiple description coders. The encoders f_s map the speech signal X to two descriptions I_s, each of rate R_s. From these two descriptions, three reconstruction values from three decoders can be obtained, depending on which combi-

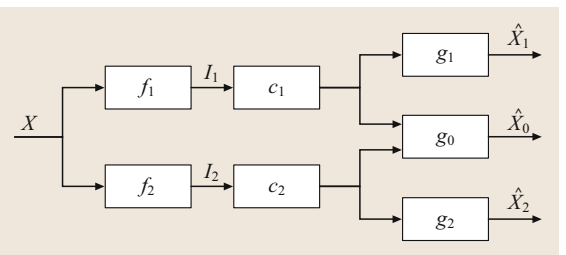


Fig. 15.12 Block diagram of a multiple description system with two channels c_s, two encoders f_s, and three decoders g_0 and g_s. Each channel is denoted by an index $s \in \{1, 2\}$

nation of descriptions is used. When both descriptions are received, the central decoder is active and the reconstructed value is $\hat{X}_0 = g_0(I_1, I_2)$. When only one description is received, the corresponding side decoder is active and the reconstructed value is $\hat{X}_s = g_s(I_s)$. The design problem of a multiple description system is to find encoders f_s and decoders g_0 and g_s that minimize the central distortion $E[d(X, \hat{X}_0)]$, with given constraints on side distortions $E[d(X, \hat{X}_s)] < D_s$, for all combinations of rates R_s. The region of interest in $(R_1, R_2, D_0, D_1, D_2)$ is the performance region, which consists of all achievable combinations.

Bounds on MDC Performance

For the single channel case, the rate-distortion theorem in rate-distortion theory, e.g., [15.39], provides the achievable rates for a given distortion. For the case of multiple descriptions with the given bounds D_s on each side distortion, the achievable rates for the side descriptions are

$$R_s \geq I(X; \hat{X}_s) \,, \tag{15.5}$$

for each channel $s \in \{1, 2\}$, where $I(X; \hat{X})$ is the mutual information between X and $\hat{X}$.

For the central description, when both channels are in the working state, the total rate is bounded as

$$R_1 + R_2 \geq I(X; \hat{X}_1, \hat{X}_2) + I(\hat{X}_1; \hat{X}_2) \,, \tag{15.6}$$

according to *El Gamal* and *Cover* in [15.40]. Unfortunately, this bound is usually loose.

Interpretation of the bounded multiple description region is straightforward. Any mutual information between the side descriptions increases the total rate of the system. If the design is such that no mutual information exists between the two side descriptions, then

$$R_1 + R_2 \geq I(X; \hat{X}_1, \hat{X}_2) = I(X; \hat{X}_1) + I(X; \hat{X}_2) \,, \tag{15.7}$$

and the effective rate is maximized. In this case, all redundancy is lost, which leads to the minimum possible central distortion $D_0 = D(R_1 + R_2)$.

In the other extreme, maximizing redundancy gives two identical descriptions and

$$R_1 + R_2 \geq 2I(X; \hat{X}_s) \,. \tag{15.8}$$

However, this increases the central distortion to $D_0 = D(R_s) = D_s$ and nothing is gained in terms of distortion. Note that this is the same setup as the simple FEC method in the beginning of the section on Media-Dependent FEC in Sect. 15.4.1.

Bounds for Gaussian Sources

The bounds of the multiple description region are, as stated earlier, generally loose and the region is not fully known. Only for the case of memoryless Gaussian sources and the squared error distortion criterion is the region fully known. While this distribution is not representative of the speech itself, the result provides insight to the performance of multiple description coding. *Ozarow* showed in [15.41] that the region defined by the bounds of El Gamal and Cover are tight in this case and define all achievable combinations of rates and distortions. The bounds on distortions as functions of rate are

$$D_s \geq \sigma^2 2^{-2R_s} \tag{15.9}$$

$$D_0 \geq \sigma^2 2^{-2(R_1+R_2)} \cdot \gamma_D(R_1, R_2, D_1, D_2) \,, \tag{15.10}$$

where the variance σ^2 is used and

$$\gamma_D = \begin{cases} 1 & \text{if } D_1 + D_2 > \sigma^2 + D_0 \,, \\ \frac{1}{1-a^2} & \text{otherwise} \,, \end{cases} \tag{15.11}$$

$$a = \sqrt{(1-D_1)(1-D_2)} - \sqrt{D_1 D_2 - 2^{-2(R_1+R_2)}} \,. \tag{15.12}$$

Interpretation of the bounds on side distortion is trivial. The central distortion is increased by a factor of γ_D for low side distortions. This implies that the best achievable distortion for rate $R_1 + R_2$, i. e., $D(R_1 + R_2)$, is obtained

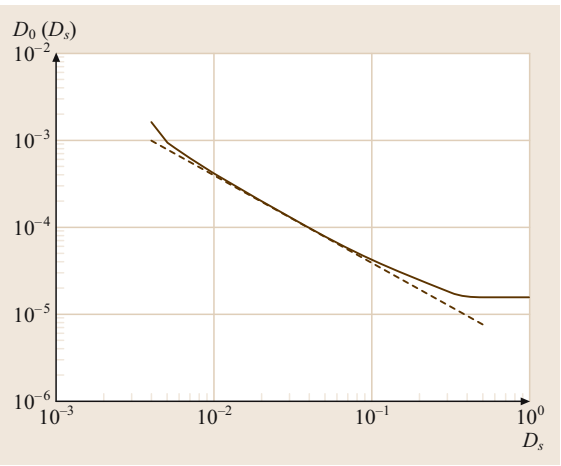


Fig. 15.13 Central distortion D_0 as a function of the side distortion D_s for a Gaussian i.i.d. source with unit variance and fixed per channel rate $R = 4$. Note the trade-off between side and central distortion. The *dashed line* represents the high-rate approximation

only if side distortions are allowed to be large. The relationship is illustrated in a plot of D_0 as a function of $D_s = D_1 = D_2$ for a fixed $R = R_1 = R_2$, in Fig. 15.13.

A high rate approximation to the bound derived by Ozarow is derived in [15.42] and is for the unit-variance Gaussian case given by

$$D_0 D_s = \frac{1}{4} 2^{-4R} . \tag{15.13}$$

MDC Versus Channel Splitting

In modern MDC methods, the side distortions decrease with increasing rate. A technique relevant to speech that does not have this property but is generally included in discussions of MDC is channel splitting by odd-even separation. In this approach odd and even channels are transmitted in separate packets, and if a packet is lost interpolation (in some cases with the aid of an additional coarse quantizer) is used to counter the effect of the loss. Variants on this approach are found in [15.43–46] and a discussion on early unpublished work on this topic can be found in [15.47]. The method is perhaps more accurately classified as a Wyner–Ziv-type coding method (or distributed source coding method) [15.48, 49] with the odd and the even samples forming correlated sources that are encoded separately. In general, the performance of the odd–even separation based methods are suboptimal. The reason is that the redundancy between descriptions is highly dependent of the redundancy present in the speech signal. Hence, the trade-off between the side distortion and the central distortion is difficult to control.

MDC Scalar Quantization

Two methods for design of multiple description scalar quantization, resolution-constrained and entropy-constrained, were described by *Vaishampayan* in [15.50] and [15.51]. In resolution-constrained quantization, the codeword length is fixed and its resolution is constrained. In entropy-constrained quantization the codeword length is variable and the index entropy is constrained. Vaishampayan assumes the source to be represented by a stationary and ergodic random process X. The design of the resolution-constrained coder is described next, followed by a description of the entropy-constrained coder.

The encoder first maps a source sample x to a partition cell in the central partition $\mathcal{A} = \{A_1, \ldots, A_N\}$. The resulting cell with index i_0 in $\mathcal{A}$ is then assigned two indices via the index assignment function $a(i_0) = \{i_1, i_2\}$, where $i_s \in \mathcal{I}_s = \{1, 2, \ldots, M_s\}$ and $N \le M_1 M_2$. The index assignment function is such that an inverse $i_0 = a^{-1}(i_1, i_2)$ referring to cell A_{i_0} always exists. The two indices, resolution-constrained, are sent over each channel to the receiver. As is appropriate for constrained-resolution coding, the codewords are of equal length.

At the receiver end, depending on which channels are in a working state, a decoder is engaged. In the case when one of the channels is not functioning, the received index of the other channel is decoded by itself to a value $\hat{x}_s = g_s(i_s)$ in the corresponding codebook $\hat{\mathcal{X}}_s = \{\hat{x}_{s,1}, \hat{x}_{s,2}, \ldots, \hat{x}_{s,M_s}\}$. In the case when both channels deliver an index, these are used to form the central reconstruction value $\hat{x}_0 = g_0(i_1, i_2)$ in codebook $\hat{\mathcal{X}}_0 = \{\hat{x}_{0,1}, \hat{x}_{0,2}, \ldots, \hat{x}_{0,N}\}$. The three decoders are referred to as $\boldsymbol{g} = \{g_0, g_1, g_2\}$.

The performance of the outlined coder is evaluated in terms of a Lagrangian, $L(\mathcal{A}, \boldsymbol{g}, \lambda_1, \lambda_2)$, which is dependent on the choice of partition $\mathcal{A}$, decoders $\boldsymbol{g}$, and Lagrange multipliers λ_1 and λ_2. The Lagrange multipliers weight the side distortions. The codeword length constraints are implicit in the partition definition and do not appear explicitly in the Lagrangian. Minimizing the Lagrangian L is done with a training algorithm that is based on finding non-increasing values of L by iteratively holding $\mathcal{A}$ and $\boldsymbol{g}$ constant while optimizing for $\boldsymbol{g}$ and $\mathcal{A}$, respectively.

The performance of the training algorithm presented is highly dependent on the index assignment $a(i_0) = \{i_1, i_2\}$. This mapping should be such that it minimizes the distortion for given channel rates $R_s = \log_2(M_s)$. The minimization is done by comparing all possible combinations of indices i_1 and i_2 for all possible cardinalities of $\mathcal{A}$, $N \le M_1 M_2$. This kind of search is too complex, as the total number of possible index assignments is $\sum_{N=1}^{M_1 M_2} (M_1 M_2)!/(M_1 M_2 - N)!$. A suboptimal search algorithm is presented in [15.52]. An illustration of an index assignment that is believed

	1	2	3	4	5	6	7	8
1	1	3						
5	2	4	5					
3		6	7	9				
4			8	10	11			
5				12	13	15		
6					14	16	17	
7						18	19	21
8							20	22

Fig. 15.14 Nested index assignment matrix for $R_s = 3$ bits. Vertical and horizontal axis represent the first and second channel, respectively

to be close to optimal, the *nested index assignment* [15.50], is found in Fig. 15.14. There, the index assignment matrix shows the procedure of inverse mapping $i_0 = a^{-1}(i_1, i_2)$. It is clear that the redundancy in the system is directly dependent on N. The more cells the central partition has, the lower central distortion is achievable at the expense of side distortions.

The performance of resolution-constrained multiple description scalar quantization was analyzed for high rates in [15.42]. It was shown that the central and side distortions for a squared distortion criterion are dependent of the source probability distribution function $p(x)$ as

$$D_0 = \frac{1}{48}\left(\int_{-\infty}^{\infty} p^{1/3}(x)\,\mathrm{d}x\right)^3 2^{-2R(1+a)} \qquad (15.14)$$

$$D_s = \frac{1}{12}\left(\int_{-\infty}^{\infty} p^{1/3}(x)\,\mathrm{d}x\right)^3 2^{-2R(1-a)}\,, \qquad (15.15)$$

where $a \in (0, 1)$. Writing this interdependency as the product of the distortions gives better understanding, since the equation then is independent of a. For the special case of a unit-variance Gaussian source we have

$$D_0 D_s = \frac{1}{4}\left(\frac{2\pi\,\mathrm{e}}{4\,\mathrm{e}3^{-1/2}}\right)^2 2^{-4R}\,. \qquad (15.16)$$

The gap of this high-rate scalar resolution-constrained quantizer to the rate-distortion bound computed by Ozarow is 8.69 dB.

The design of the entropy-constrained coder resembles the resolution-constrained coder described above. The differences are the replacement of fixed codeword length constraints by entropy constraints on index entropies, as well as added variable-length coders prior to transmission over each channel.

The Lagrangian function for the entropy-constrained case, $L(\mathcal{A}, \boldsymbol{g}, \lambda_1, \lambda_2, \lambda_3, \lambda_4)$, depends on six variables. These are the partition $\mathcal{A}$, decoders $\boldsymbol{g}$ and Lagrange multipliers λ_1 through λ_4, where two of the Lagrangian multipliers correspond to the constraints on the index entropies. Thus, in this case the rate constraints are explicitly in the Lagrangian. The Lagrangian is, as for the resolution-constrained case, minimized with a training algorithm. Before sending indices over each channel, variable length coders are applied to indices obtained by the index assignment.

Disregarding the requirement that codewords should have equal length results in improved performance. The high-rate approximation of the product between central and side distortions for the unit-variance Gaussian case is

$$D_0 D_s = \frac{1}{4}\left(\frac{2\pi\,\mathrm{e}}{12}\right)^2 2^{-4R}\,. \qquad (15.17)$$

The resulting gap of the scalar constrained-entropy quantizer to the rate-distortion bound is here 3.07 dB.

MDC Vector Quantization

It is well known from classical quantization theory that the gap between the rate-distortion bound and the performance of the scalar quantizer can be closed by using vector quantization. The maximum gain of an optimal vector quantizer over an optimal scalar quantizer due to the space filling advantage is 1.53 dB [15.53]. This motivates the same approach in the case of multiple descriptions. One approach to multiple description vector quantization is described in the following. It uses lattice codebooks and is described in [15.54], where implementation details are provided for the A_2 and $\mathbb{Z}_i$ lattices, with $i = 1, 2, 4, 8$.

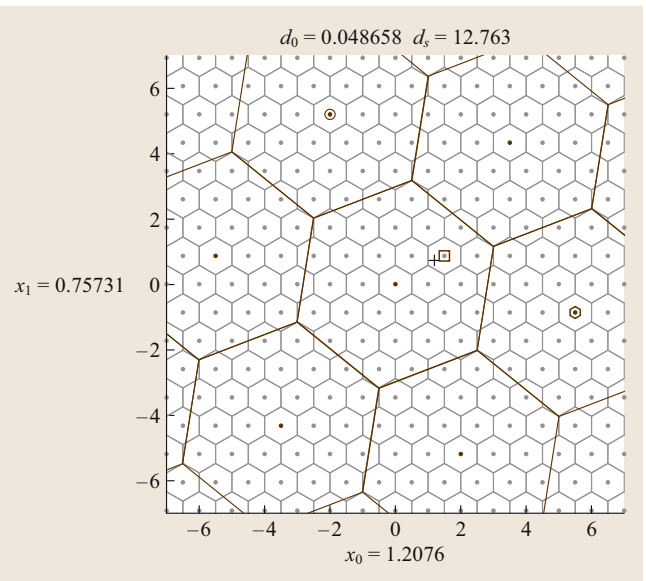


Fig. 15.15 Encoding of the vector marked with a cross. The two descriptions are marked with brown circle and hex. When both descriptions are received the reconstruction point is chosen as the brown square. Central and side distortions are shown at the top

Multiple description vector quantization with lattice codebooks is a method for encoding a source vector $\boldsymbol{x} = [x_1, \ldots, x_n]^{\mathrm{T}}$ with two descriptions. In [15.54], an algorithm for two symmetric descriptions, equal side distortions, is described. These two descriptions are, as with multiple description scalar quantization, sent over different channels, providing a minimum fidelity in the case of failure of one of the channels. Figure 15.15 illustrates the encoded and decoded codewords for a two-dimensional source vector $\boldsymbol{x}$. The finer lattice Λ represents the best resolution that can be obtained, that is with the central reconstruction value. Codewords that are sent over the channel are actually not a part of the fine lattice Λ, but of a geometrically similar sublattice Λ'. The sublattice Λ' is obtained by scaling, rotating, and possibly reflecting Λ. This procedure is dependent on what trade-off one wants to obtain between central and side distortions. Which points in Λ' to send and how to choose a reconstruction point in Λ given these points is called the labeling problem, which is comparable to the index assignment problem in the one dimensional case. The labeling problem is solved by setting up a mapping $\Lambda \rightarrow \Lambda' \times \Lambda'$ for all cells of Λ that are contained within the central cell of Λ'. This mapping is then extended to all cells in Λ using the symmetry of the lattices.

Choosing distortions is done by scaling of the used lattices. However, the constraint that the side-distortions must be equal is a drawback of the method. This is improved in [15.55], where an asymmetric multiple description lattice vector quantizer is presented.

High-rate approximations for the setup described above with coding of infinitely large vectors result in a product of distortions given by

$$D_0 D_s = \frac{1}{4}\left(\frac{2\pi\,\mathrm{e}}{2\pi\,\mathrm{e}}\right)^2 2^{-4R} \tag{15.18}$$

for a unit-variance Gaussian source.

Looking closer at the equation, the gain compared to the approximation of the rate-distortion bound is 0 dB. This means that the high-rate approximation of the rate-distortion bound has been reached. In an implementation, however, finite-dimension vectors must be used. For the example provided above, with the hexagonal A_2 lattice, the distortion product is

$$D_0 D_s = \frac{1}{4}\left(\frac{2\pi\,\mathrm{e}}{5/36\sqrt{3}}\right)^2 2^{-4R} \tag{15.19}$$

and a gap of 1.53 dB remains compared to the approximation of the rate distortion bound. Using higher dimensions results in greater gains.

Correlating Transforms

Multiple description coding with correlating transforms was first introduced by *Wang*, *Orchar*, and *Reibman* in [15.56]. Their approach to the MDC problem is based on a linear transformation of two random variables, which introduces correlation in the transform coefficients. This approach was later generalized by *Goyal* and *Kovacevic* in [15.57] to M descriptions of an N-dimensional vector, where $M \leq N$.

Consider a vector $\boldsymbol{y} = [y_1, y_2]^{\mathrm{T}}$ with variances σ_1 and σ_2. Transforming the vector with an orthogonal transformation such as

$$\mathbf{z} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\boldsymbol{y}$$

produces transform coefficients $\mathbf{z} = [z_1, z_2]^{\mathrm{T}}$. Sending one transform coefficient over each channel results in central distortion

$$D_0 = \frac{\pi\,\mathrm{e}}{6}\left(\frac{\sigma_1^2+\sigma_2^2}{2}\right)2^{-2R} \tag{15.20}$$

and average side distortion

$$D_s = \frac{\sigma_1^2\sigma_2^2}{\sigma_1^2+\sigma_2^2} + \frac{\pi\,\mathrm{e}}{12}\left(\frac{\sigma_1^2+\sigma_2^2}{2}\right)2^{-2R} \tag{15.21}$$

on vector $\boldsymbol{y}$. When interpreting the results, we see that the relation of σ_1 and σ_2 determines the trade-off between central and side distortion. If σ_1 equals σ_2, the distortions obtained are equal to what would have been obtained if $\boldsymbol{y}$ was sent directly, i. e., the case with no redundancy between descriptions.

The example above leads to the conclusion that a transformation on the i.i.d. source vector $\boldsymbol{x}$ needs to be performed to get transform coefficients $\boldsymbol{y}$ of different variances. These are then handled as described above.

Extension to an N-dimensional source vector $\boldsymbol{x}$ is handled in the following manner. A square correlating transform produces N coefficients in $\boldsymbol{y}$ with varying variances. These coefficients are quantized and transformed to N transform coefficients, which in turn are placed into M sets in an a priori fixed manner. These M sets form the M descriptions. Finally, entropy coding is applied to the descriptions.

High-rate approximations of multiple description coding with correlating transforms does not tell us anything about the decay of distortion. This is so since

independent of choice of transform $D_0 = O(2^{-2R})$ and $D_1 = D_2 = O(1)$. Simulations [15.57] show however that multiple description coding with correlating transforms performs well at low redundancies.

Conclusions on MDC

This section has presented the bounds on achievable rate-distortion for multiple description coding and a number of specific algorithms have been discussed. We provided some useful comparisons.

Figure 15.16 shows an overview of the performance of the discussed algorithms for i.i.d. Gaussian signals and the squared error criterion. The figure is based on the high-rate approximations of achievable side and central distortions of resolution-constrained multiple description scalar quantization, entropy-constrained multiple description scalar quantization and multiple description vector quantization with dimension two. It is clear that the distortions achievable decrease in the order that the methods are mentioned. Using vectors of dimension larger than two reduces the distortions further, approaching the high-rate approximation of the Ozarow bound for vectors of infinite dimension. Note that multiple description with correlating transforms is not included in the figure, due to the reasons mentioned in the previous section.

Evaluating the performance of the three methods proposed by *Vaishampayan* [15.50, 51, 54], i. e., constrained-resolution MDC scalar quantization, constrained-entropy MDC scalar quantization and MDC vector quantization with lattice codebooks for constrained-entropy coding, the results are not surprising. They are consistent with what is known in equivalent single description implementations. Entropy-constrained quantization performs better than resolution-constrained quantization because the average rate constraint is less stringent than the fixed rate constraint. Even for the i.i.d. case, vector quantization outperforms scalar quantization because of the space filling and shape (constrained resolution only) advantages [15.58].

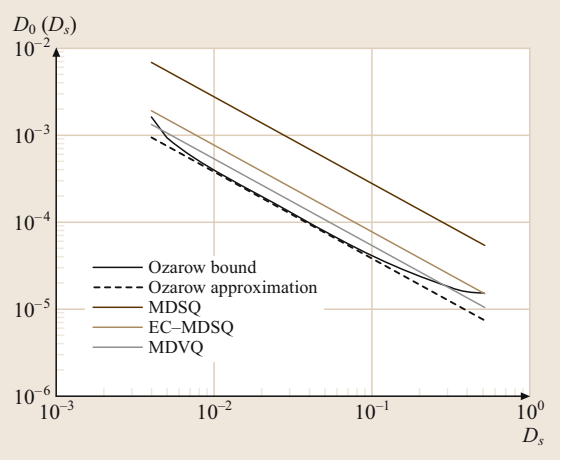


Fig. 15.16 A comparison of the Ozarow bound, its high-rate approximation, and the high-rate approximations of three coders

Improved performance has a price. While the quantization step is often of lower computational complexity for entropy-constrained quantization, it requires an additional lossless encoding and decoding step. The drawbacks of vector quantization are that it introduces delay in the signal and increases computational effort. For constrained-entropy quantizers the added cost generally resides in the variable-length encoding of the quantization indices. For constrained-resolution coding the codebook search procedure generally increases rapidly in computational effort with increasing vector dimension.

The usage of correlating transforms to provide multiple descriptions has been shown to work well for low redundancy rates [15.57]. However, better performance is obtained with the approaches using an index assignment when more redundancy is required. Hence, one might choose not to implement correlating transforms even for low-redundancy applications to avoid changes in the coder design if additional redundancy is required at a later time.

When it comes to speech applications, the described multiple description coding techniques are in general not directly applicable. The speech source is not i.i.d., which was the basic assumption in this section. Thus, the assumption that speech is an i.i.d. source is associated with performance loss. Only pulse-code modulation (PCM) systems are based on the i.i.d. assumption, or perhaps more correctly, ignore the memory that is existent in speech. It is straightforward to adapt the PCM coding systems to using the described multiple description theory, and a practical example is [15.59]. Other speech coding systems are generally based on prediction [e.g., differential PCM (DPCM), adaptive DPCM (ADPCM), and code excited linear prediction (CELP)] making the application of MDC less straightforward. The MDC theory can be applied to the encoding of the prediction parameters (if they are transmitted as side information) and to the excitation. However, the prediction loops at encoder and decoder are prone to mismatch and, hence, error propagation. An alternative approach to exploiting the memory of the source has

been described in [15.60]. A Karhunen–Loève transform is applied to decorrelate a source vector prior to usage of MDC. However, since transform coding is not commonly used in speech coders, this method is not applicable either.

Multiple description theory has made large advances in the last two decades. The results in artificial settings are good. However, much work remains to be done before its promise is fulfilled in practical speech coding applications.

15.5 Packet Loss Concealment

If no redundancy is added at the encoder and all processing to handle packet loss is performed at the decoder the approach is often referred to as packet loss concealment (PLC), or more generally error concealment. Sometimes, a robust encoding scheme is combined with a PLC, where the latter operates as a safety net, set up to handle lost packets that the former did not succeed in recovering.

Until recently, two simple approaches to dealing with lost packets have prevailed. The first method, referred to as zero stuffing, involves simply replacing a lost packet with a period of silence of the same duration as the lost packet. Naturally, this method does not provide a high-quality output and, already for packet loss rates as low as 1%, very annoying artifacts are apparent. The second method, referred to as packet repetition, assumes that the difference between two consecutive speech frames is quite small and replaces the lost packet by simply repeating the previous packet. In practice, it is virtually impossible to achieve smooth transitions between the packets with this approach. The ear is very sensitive to discontinuities which leads to audible artifacts as a result of the discontinuities. Furthermore, even a minor change in pitch frequency is easily detected by the human ear. However, this approach performs fairly well at low packet loss probabilities (less than 3%).

More-advanced approaches to packet loss design are based on signal analysis where the speech signal is extrapolated or interpolated to produce a more natural-sounding concealment. These approaches can be divided into two basic classes: nonparametric and parametric methods. If the jitter buffer contains one or more future packets in addition to the past packets, the missing signal can be generated by interpolation. Otherwise, the waveform in the previous frame is extrapolated into the missing frame. Two-sided interpolation generally gives better results than extrapolation.

15.5.1 Nonparametric Concealment

Overlap-and-add (OLA) techniques, originally developed for time-scale modification, belong to the class of nonparametric concealment methods used in VoIP. The missing frame is in OLA generated by time-stretching the signal in the adjacent frames. The steps of the basic OLA [15.61] are:

1. Extract regularly spaced windowed speech segments around sampling instants $\tau(kS)$.
2. Space windowed segments according to time scaling (regularly spaced S samples apart).
3. Normalize the sum of segments as

$$y(n) = \frac{\sum_k v(n-kS)x(n+\tau(kS)-kS)}{\sum_k v(n-kS)} , \quad (15.22)$$

where $v(n)$ is a window function. Due to the regular spacing and with a proper choice of symmetric window the denominator becomes constant, i. e., $\sum_k v(n-kS) = C$, and the synthesis is thus simplified.

OLA adds uncorrelated segments with similar short-term correlations, thus retaining short-term correlations. However, the pitch structure, i. e., spectral fine structure, has correlations that are long compared to the extraction window and these correlations are destroyed, resulting in poor performance. Two significant improvement approaches are synchronized OLA (SOLA) [15.62] and waveform similarity OLA (WSOLA) [15.63].

In SOLA the extracted windowed speech segments are regularly spaced as in OLA, but when spacing the output segments they are repositioned such that they have high correlation with the already formed portion. The generated segment is formed as

$$y(n) = \frac{\sum_k v(n-kS+\delta_k)x(n+\tau(kS)-kS+\delta_k)}{\sum_k v(n-kS+\delta_k)} . \quad (15.23)$$

A renormalization is needed after placing the segment due to the nonconstant denominator. The window shift δ_k is searched and selected such that the cross-correlation

between the windowed segment $v(n-kS+\delta_k)x(n+\tau(kS)-kS+\delta_k)$ and the previously generated output

$$y_{k-1}(n) = \frac{\sum_{m=-\infty}^{k-1} v(n-mS+\delta_m)x(n+\tau(mS)-mS+\delta_m)}{\sum_{m=-\infty}^{k-1} v(n-mS+\delta_m)} \quad (15.24)$$

is maximized.

WSOLA, instead, extracts windowed segments that are selected for maximum cross-correlation with the last played out segment and regularly space these at the output. The constant denominator implies that there is no need to re-normalize and WSOLA is thus simpler than SOLA.

$$y(n) = \frac{\sum_k v(n-kS)x(n+\tau(kS)-kS+\delta_k)}{\sum_k v(n-kS)} = \sum_k v(n-kS)x(n+\tau(kS)-kS+\delta_k)\,. \quad (15.25)$$

An example of PLC using WSOLA is presented in [15.64].

The two techniques can efficiently be utilized in compensating for packet delays, as well as for pure PLC as mentioned in Sect. 15.3.5. For example, if a packet is not lost and only delayed less than a full frame interval the signal can be stretched this time amount until the play-out of the delayed frame starts. Examples of this are [15.25] for SOLA and [15.24] for WSOLA.

15.5.2 Parametric Concealment

Waveform substitution methods were early PLC methods that tried to find a pitch cycle waveform in the previous frames and repeat it. *Goodman* et al. [15.65] introduced two approaches to waveform substitution, a pattern matching approach and a pitch detection approach. The pattern matching approach used the last samples of the previous frame as a template and searched for a matching segment earlier in the received signal. The segment following this match was then used to generate the missing frame. The pitch detection approach estimated the pitch period and repeated the last pitch cycle. According to [15.66], the pitch detection approach yielded better results.

Voicing, power spectrum and energy are other example of features, besides the pitch period, that can be estimated for the previous speech segments and extrapolated in the concealed segment. The packet loss concealment method for the waveform codec G.711, as specified in annex I [15.67] to the ITU standard, is an example of such a method.

If the PLC is designed for a specific codec and has access to the decoder and its internal states, better performance can generally be obtained than if only the decoded waveform is available. Many codecs, such as G.729, have a built-in packet loss concealment algorithm that is based on knowledge of the decoder parameters used for extrapolation.

Recent approaches to PLC include different types of speech modeling such as linear prediction [15.68], sinusoidal extrapolation [15.69], multiband excitation [15.70], and nonlinear oscillator models [15.71]. The latter uses an oscillator model of the speech signal, consisting of a transition codebook built from the existing signal, which predicts future samples. It can advantageously be used for adaptive play-out similar to the OLA methods. Further, in [15.72], a PLC method is presented for LPC-based coders, where the parameter evolution in the missing frames is determined by hidden Markov models.

Concealment systems that are not constrained by producing an integer number of whole frames, thus becoming more flexible in the play-out, have the potential to produce higher-quality recovered speech. This has been confirmed by practical implementations [15.23].

15.6 Conclusion

Designing a speech transmission system for VoIP imposes many technical challenges, some similar to those of traditional telecommunications design, and some specific to VoIP. The most important challenges for VoIP are a direct result of the characteristics of the transport media — IP networks. We showed that overall delay of such networks can be a problem for VoIP and that, particularly for the small packets common in VoIP, significant packet loss often occurs at networks loads that are far below the nominal capacity of the network. It can be concluded that, if a VoIP system is to provide the end user with low transmission delay and with high voice quality, it must be able to handle packet loss and transmission time jitter efficiently.

In this chapter, we discussed a number of techniques to address the technical challenges imposed by the network on VoIP. We concluded that, with proper design, it is generally possible to achieve VoIP voice quality that is equal or even better than PSTN. The extension of the signal bandwidth to 8 kHz is a major contribution towards improved speech quality. Multiple description coding is a powerful technique to address packet loss and delay in an efficient manner. It has been proven in practical applications and provides a theoretical framework that facilitates further improvement. Significant contributions towards robustness and minimizing overall delay can also be made by the usage of adaptive jitter buffers that provide flexible packet loss concealment. The combination of wide-band, multiple description coding, and packet loss concealment facilitates VoIP with high speech quality and a reasonable latency.

References

15.1 L.R. Rabiner, R.W. Schafer: *Digital Processing of Speech Signals* (Prentice Hall, Englewood Cliffs 1978)

15.2 W. Stallings: *High-Speed Networks: TCP/IP and ATM Design Principles* (Prentice Hall, Englewood Cliffs 1998)

15.3 Information Sciences Institute: Transmission control protocol, IETF **RFC793** (1981)

15.4 J. Postel: User datagram protocol, IETF **RFC768** (1980)

15.5 H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson: RTP a transport protocol for real-time applications, IETF **RFC3550** (2003)

15.6 ITU-T: G.131: Talker echo and its control (2003)

15.7 ITU-T: G.114: One-way transmission time (2003)

15.8 C.G. Davis: An experimental pulse code modulation system for short haul trunks, Bell Syst. Tech. J. **41**, 25–97 (1962)

15.9 IEEE: 802.11: Part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications (2003)

15.10 IEEE: 802.15.1: Part 15.1: Wireless medium access control (MAC) and physical layer (PHY) specifications for wireless personal area networks (WPANs) (2005)

15.11 E. Dimitriou, P. Sörqvist: Internet telephony over WLANs, 2003 USTAs Telecom Eng. Conf. Supercomm (2003)

15.12 ITU-T: G.711: Pulse code modulation (PCM) of voice frequencies (1988)

15.13 IEEE: 802.1D Media access control (MAC) bridges (2004)

15.14 D. Grossman: New terminology and clarifications for diffserv, IETF **RFC3260** (2002)

15.15 R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin: Resource ReSerVation Protocol (RSVP) – Version 1 Functional specification, IETF **RFC2205** (1997)

15.16 C. Aurrecoechea, A.T. Campbell, L. Hauw: A survey of QoS architectures, Multimedia Syst. **6**(3), 138–151 (1998)

15.17 IEEE: 802.11e: Medium Access Control (MAC) Quality of Service (QoS) Enhancements (2005)

15.18 E. Hänsler, G. Schmidt: *Acoustic Echo and Noise Control – A Practical Approach* (Wiley, New York 2004)

15.19 ITU-T: G.729: Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear prediction (CS-ACELP) (1996)

15.20 S. Andersen, A. Duric, H. Astrom, R. Hagen, W.B. Kleijn, J. Linden: Internet Low Bit Rate Codec (iLBC), IETF **RFC3951** (2004)

15.21 Ajay Bakre: www.globalipsound.com/datasheets/isac.pdf (2006)

15.22 S.B. Moon, J.F. Kurose, D.F. Towsley: Packet audio playout delay adjustment: Performance bounds and algorithms, Multimedia Syst. **6**(1), 17–28 (1998)

15.23 Ajay Bakre: www.globalipsound.com/datasheets/neteq.pdf (2006)

15.24 Y. Liang, N. Farber, B. Girod: Adaptive playout scheduling and loss concealment for voice communication over IP networks, IEEE Trans. Multimedia **5**(4), 257–259 (2003)

15.25 F. Liu, J. Kim, C.-C.J. Kuo: Adaptive delay concealment for internet voice applications with packet-based time-scale modification, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (2001)

15.26 ITU-T: P.800: Methods for subjective determination of transmission quality (1996)

15.27 S. Pennock: Accuracy of the perceptual evaluation of speech quality (PESQ) algorithm, Proc. Measurement of Speech and Audio Quality in Networks (2002)

15.28 M. Varela, I. Marsh, B. Grönvall: A systematic study of PESQs behavior (from a networking perspective), Proc. Measurement of Speech and Audio Quality in Networks (2006)

15.29 ITU-T: P.862: Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrowband telephone networks and speech codecs (2001)

15.30 ITU-T: P.862.1: Mapping function for transforming P.862 raw result scores to MOS-LQO (2003)

15.31 C. Perkins, O. Hodson, V. Hardman: A survey of packet loss recovery techniques for streaming audio, IEEE Network **12**, 40–48 (1998)

15.32 J. Rosenberg, H. Schulzrinne: An RTP payload format for generic forward error correction, IETF **RFC2733** (1999)

15.33 J. Lacan, V. Roca, J. Peltotalo, S. Peltotalo: Reed–Solomon forward error correction (FEC), IETF (2007), work in progress

15.34 J. Rosenberg, L. Qiu, H. Schulzrinne: Integrating packet FEC into adaptive voice playout buffer algorithms on the internet, Proc. Conf. Comp. Comm. (IEEE INFOCOM 2000) (2000) pp. 1705–1714

15.35 W. Jiang, H. Schulzrinne: Comparison and optimization of packet loss repair methods on VoIP perceived quality under bursty loss, Proc. Int. Workshop on Network and Operating System Support for Digital Audio and Video (2002)

15.36 E. Martinian, C.-E.W. Sundberg: Burst erasure correction codes with low decoding delay, IEEE Trans. Inform. Theory **50**(10), 2494–2502 (2004)

15.37 C. Perkins, I. Kouvelas, O. Hodson, V. Hardman, M. Handley, J. Bolot, A. Vega-Garcia, S. Fosse-Parisis: RTP payload format for redundant audio data, IETF **RFC2198** (1997)

15.38 J.-C. Bolot, S. Fosse-Parisis, D. Towsley: Adaptive FEC-based error control for internet telephony, Proc. Conf. Comp. Comm. (IEEE INFOCOMM '99) (IEEE, New York 1999) p. 1453-1460

15.39 T.M. Cover, J.A. Thomas: *Elements of Information Theory* (Wiley, New York 1991)

15.40 A.A.E. Gamal, T.M. Cover: Achievable rates for multiple descriptions, IEEE Trans. Inform. Theory **IT-28**(1), 851–857 (1982)

15.41 L. Ozarow: On a source coding prolem with two channels and three receivers, Bell Syst. Tech. J. **59**, 1909–1921 (1980)

15.42 V.A. Vaishampayan, J. Batllo: Asymptotic analysis of multiple description quantizers, IEEE Trans. Inform. Theory **44**(1), 278–284 (1998)

15.43 N.S. Jayant, S.W. Christensen: Effects of packet losses in waveform coded speech and improvements due to an odd-even sample-interpolation procedure, IEEE Trans. Commun. **COM-29**(2), 101–109 (1981)

15.44 N.S. Jayant: Subsampling of a DPCM speech channel to provide two self-contained half-rate channels, Bell Syst. Tech. J. **60**(4), 501–509 (1981)

15.45 A. Ingle, V.A. Vaishampayan: DPCM system design for diversity systems with applications to packetized speech, IEEE Trans. Speech Audio Process. **3**(1), 48–58 (1995)

15.46 A.O.W. Jiang: Multiple description speech coding for robust communication over lossy packet networks, IEEE Int. Conf. Multimedia and Expo (2000) pp. 444–447

15.47 V.K. Goyal: Multiple description coding: Compression meets the network, IEEE Signal Process. Mag. **18**, 74–93 (2001)

15.48 A.D. Wyner: Recent results in the Shannon theory, IEEE Trans. Inform. Theory **20**(1), 2–10 (1974)

15.49 A.D. Wyner, J. Ziv: The rate-distortion function for source coding with side information at the decoder, IEEE Trans. Inform. Theory **22**(1), 1–10 (1976)

15.50 V.A. Vaishampayan: Design of multiple description scalar quantizers, IEEE Trans. Inform. Theory **IT-39**(4), 821–834 (1993)

15.51 V.A. Vaishampayan, J. Domaszewicz: Design of entropy-constrained multiple-description scalar quantizers, IEEE Trans. Inform. Theory **IT-40**(4), 245–250 (1994)

15.52 N. Görtz, P. Leelapornchai: Optimization of the index assignments for multiple description vector quantizers, IEEE Trans. Commun. **51**(3), 336–340 (2003)

15.53 R.M. Gray: *Source Coding Theory* (Kluwer, Dordrecht 1990)

15.54 V.A. Vaishampayan, N.J.A. Sloane, S.D. Servetto: Multiple-description vector quantization with lattice codebooks: Design and analysis, IEEE Trans. Inform. Theory **47**(1), 1718–1734 (2001)

15.55 S.N. Diggavi, N. Sloane, V.A. Vaishampayan: Asymmetric multiple description lattice vector quantizers, IEEE Trans. Inform. Theory **48**(1), 174–191 (2002)

15.56 Y. Wang, M.T. Orchard, A.R. Reibman: Multiple description image coding for noisy channels by pairing transform coefficients, IEEE Workshop on Multimedia Signal Processing (1997) pp. 419–424

15.57 V.K. Goyal, J. Kovacevic: Generalized multiple description coding with correlating transforms, IEEE Trans. Inform. Theory **47**(6), 2199–2224 (2001)

15.58 T. Lookabough, R. Gray: High-resolution theory and the vector quantizer advantage, IEEE Trans. Inform. Theory **IT-35**(5), 1020–1033 (1989)

15.59 Ajay Bakre: www.globalipsound.com/datasheets/ipcm-wb.pdf (2006)

15.60 J. Batllo, V.A. Vaishampayan: Asymptotic performance of multiple description transform codes, IEEE Trans. Inform. Theory **43**(1), 703–707 (1997)

15.61 D.W. Griffin, J.S. Lim: Signal estimation from modified short-time Fourier transform, IEEE Trans. Acoust. Speech Signal Process. **32**, 236–243 (1984)

15.62 S. Roucos, A. Wilgus: High quality time-scale modification for speech, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1985) pp. 493–496

15.63 W. Verhelst, M. Roelands: An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1993) pp. 554–557

15.64 H. Sanneck, A. Stenger, K. Ben Younes, B. Girod: A new technique for audio packet loss concealment, Proc. Global Telecomm. Conf GLOBECOM (1996) pp. 48–52

15.65 D.J. Goodman, G.B. Lockhart, O.J. Wasem, W.C. Wong: Waveform substitution techniques for recovering missing speech segments in packet voice communications, IEEE Trans. Acoust. Speech Signal Process. **34**, 1440–1448 (1986)

15.66 O.J. Wasem, D.J. Goodman, C.A. Dvorak, H.G. Page: The effect of waveform substitution on the qual-

ity of PCM packet communications, IEEE Trans. Acoust. Speech Signal Process. **36**(3), 342–348 (1988)
15.67 ITU-T: G.711 Appendix I: A high quality low-complexity algorithm for packet loss concealment with G.711 (1999)
15.68 E. Gündüzhan, K. Momtahan: A linear prediction based packet loss concealment algorithm for PCM coded speech, IEEE Trans. Acoust. Speech Signal Process. **9**(8), 778–785 (2001)
15.69 J. Lindblom, P. Hedelin: Packet loss concealment based on sinusoidal extrapolation, Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 1 (2002) pp. 173–176
15.70 K. Clüver, P. Noll: Reconstruction of missing speech frames using sub-band excitation, Int. Symp. Time-Frequency and Time-Scale Analysis (1996) pp. 277–280
15.71 G. Kubin: Proc. IEEE Int. Conf. Acoust. Speech Signal Process., Vol. 1 (1996) pp. 267–270
15.72 C.A. Rodbro, M.N. Murthi, S.V. Andersen, S.H. Jensen: Hidden Markov Model-based packet loss concealment for voice over IP, IEEE Trans. Speech Audio Process. **14**(5), 1609–1623 (2006)

19. Basic Principles of Speech Synthesis

J. Schroeter

Speech synthesis enables voice output by machines or devices. Text-to-speech (TTS) synthesis does so by using text as input. Ever since the talking machine by von Kempelen in 1791 [19.1], researchers and technologists have endeavored to make machines talk. The first electronic synthesis, Homer Dudley's *Voder* (*Voice Coder*), was demonstrated at the 1939 World Fair in New York City [19.2]. Today, TTS systems enjoy wide use in assistive technologies, telecommunications, entertainment, and education. In this chapter we will review the basic principles of this technology, which serves as an introduction to later chapters that provide the reader with more-detailed information.

19.1 The Basic Components of a TTS System

A simple block diagram of a text-to-speech (TTS) system is depicted in Fig. 19.1. The frontend on the left is logically separated from the synthesizer backend on the right. The frontend works on a symbolic level, taking text as input and creating control information as its output. The synthesis backend uses the control information and renders speech output from it. In this chapter, we will use English as an example language. We also point out that, for many aspects of TTS, there is a choice of rule-based versus data-driven methods that can be applied. Finding the right mix or compromise between both extremes is one topic of ongoing research.

19.1.1 TTS Frontend

Figure 19.2 depicts the principle tasks of a TTS frontend. It provides document structure detection and text normalization, interprets text markup, and performs a linguistic analysis to produce what amounts to tagged text that then undergoes phonetic analysis to create phone-based information, and prosodic analysis to determine pitch, durations, as well as assigning stresses and pauses. Although presented here in a simple linear sequence of steps, many of the tasks of a TTS frontend actually rely on specific information created in *later* steps. Therefore, the sequence of steps presented here is only the most commonly used with different interdependencies handled as exceptions and/or resulting in delayed decisions. Much more detailed information on text processing in TTS frontends can be found in Chap. 22 by Sproat.

Document structure detection can be as general as interpreting punctuation marks, may include *filtering* out email headers (e.g., in a unified messaging application), or may even take paragraph formatting into account. Document structure detection is simplified in case the document follows the standard generalized markup language (SGML) standard. SGML is an international standard for the definition of device-independent, system-independent methods of representing texts in electronic form [19.3].

Text normalization handles abbreviation and acronyms with the goal of matching how an educated human reader would render the input text. 'St.' could be

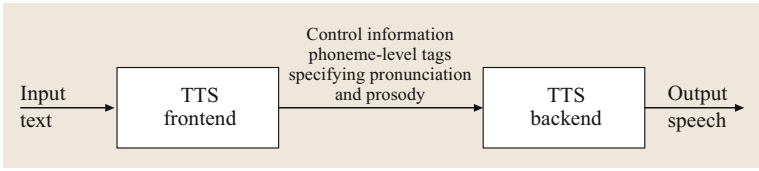


Fig. 19.1 Block diagram of a text-to-speech (TTS) system

rendered as *Street* or as *Saint*, 'Dr.' as *Drive* or *Doctor*. 'IBM' is spelled out, while 'NASDAQ' is not.

Text markup can be used to control how the TTS engine renders its output. Proper markup may make the TTS output sound *intelligent*, such as letting the frontend know that it should use address mode for reading a street address. It may also tell the TTS engine to render a sentence with emotions like *angry*, *sad*, *happy*, or *neutral*. One widely used markup standard for synthesis is speech synthesis markup language (SSML) [19.4].

In addition to text preprocessing, linguistic–syntactic analysis tasks include a morphological analysis for proper word pronunciation and a syntactic analysis to facilitate accenting and phrasing and to handle ambiguities in the written text. Most TTS systems forego fully parsing the input text to limit computational complexity and because input consists often of incomplete sentence fragments.

Phonetic analysis focuses on the phone level within each word, tagging each phone with information about what sound to produce and how to produce it (speaking style, emphasis). As part of this process, it performs a grapheme-to-phoneme conversion (i. e., determining the exact pronunciation of each word of the input sentence) and homograph disambiguation (e.g., figuring out whether an input sentence uses the present tense or the past-tense version of the word *read*). For these tasks, it relies on dictionaries to look up word pronunciations, usually backed up by general letter-to-sound rules as a fallback or, for some languages, as the primary source for pronunciations. An etymological analysis helps provide clues about the meaning of words and explain irregularities in pronunciation, for example, in names. All of this information then drives the prosodic analysis.

Prosodic analysis determines the progression of intonation, speaking rate, and loudness across an utterance, which are ultimately represented at the phoneme level as pitch (fundamental frequency), duration, and amplitude. The intonation and timing parts largely determine the rhythm of a synthetic sentence in addition to syllabic stress (expressing the prominence of a syllable relative to its surrounding syllables). Repetition of rhythm and stress patterns across sentences contributes to listener fatigue and low perceptual scores. Therefore, proper prosody facilitates listener focus, highlights

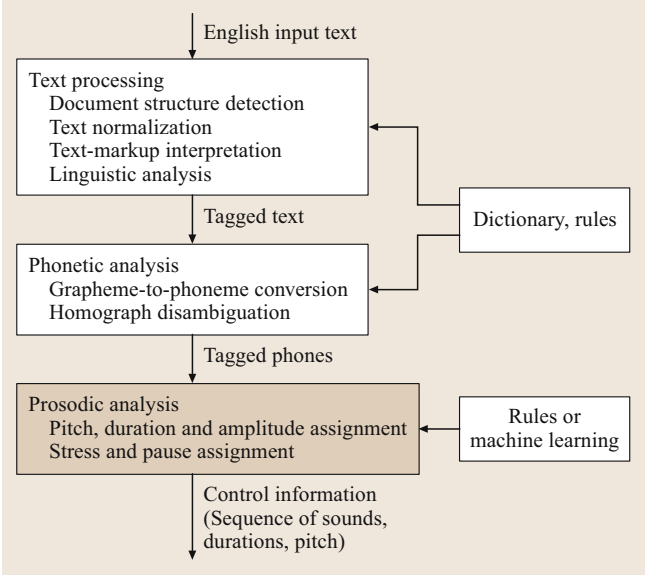


Fig. 19.2 Tasks and processing in a TTS frontend

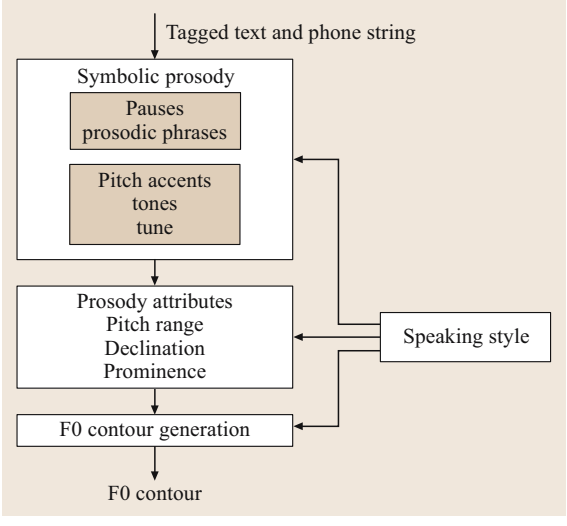


Fig. 19.3 Block diagram of determining the pitch contour of a speech utterance

important or new content, or even conveys nuances of meaning.

Timing is an important part of prosody assignment, including durations of individual phones. There are several approaches to duration modeling, but two guiding principle approaches exist. The more common rule-based approach is to start with segment intrinsic durations and moderate these durations based on rules that combine a relatively small set of different types of data, extracted by prior analyses either additively or multiplicatively [19.5, 6]. A more-recent data-driven approach is to use large speech corpora and to apply machine learning methods involving a much larger set of control factors [19.7].

Intonation refers to the tonal/melodic parts of prosody. Given a certain speaking style, Fig. 19.3 summarizes the process of intonation/pitch assignment in terms of creating an F0 contour for a given utterance. Starting from a symbolic representation, for example in terms of tone and break indices (ToBI) [19.8], pauses and prosodic phrases are determined, followed by setting pitch accents and/or tones. Suprasegmental prosodic attributes like pitch range (the difference between the highs and lows in the voice of a speaker), declination (the gradual lowering of pitch over the course of a sentence), and the expression of prominence are certainly affected by the particular speaker's speaking style.

The final step towards arriving at an F0 contour uses the available symbolic prosody data to create a time series for F0. Stylized or actually observed pitch templates in a corpus might be used. A simpler approach is to use a generative model such as Fujisaki's [19.9] that expands pulse or step-like accent and phrase commands into continuous pitch contours, employing a source–filter model. More information on prosodic processing can be found in Chap. 23.

Finally, all of the frontend information is then forwarded to the TTS backend for synthesis. Note that TTS frontends have to be somewhat language specific. Pronunciation dictionaries and letter-to-sound rules used in the frontend may even be tailored to match the specific speaker's language accent and speech production peculiarities of the human voice that TTS tries to mimic.

19.1.2 TTS Backend

TTS backends are largely independent of TTS frontends. This gives TTS system designers the option to pick from different frontends and backends, provided the interface between them can be instantiated.

A TTS backend uses the information provided by the frontend to synthesize speech using a specific synthesis method. Traditionally, we distinguished *rule-based* versus *corpus-based* synthesis methods. The former included articulatory synthesis and formant synthesis, while the latter was almost synonymous with concatenative synthesis. Note, however, that one should keep the paradigm of knowledge representation (rule versus corpus) separate from the synthesis method itself. The reason for this is that the clear-cut distinction no longer exists. We now have examples of articulatory and/or formant synthesizers that incorporate data-driven methods and other examples of using rules in concatenative synthesize backends. However, it is still true that every given synthesis method is somewhat dependent on the speech representation it uses. We will present details on speech representations in Sect. 19.2 below. First, however, we will review the principle differences between articulatory, formant, and concatenative synthesis methods.

Articulatory Synthesis

Articulatory synthesis uses mechanical and acoustic models of speech production to synthesize speech. These models can be as intricate as solving the Navier–Stokes partial differential equation for the vocal tract and the glottis [19.10].

The process of computing vocal tract acoustics from its geometry is sketched in Fig. 19.4. The dynamic movements of the articulators (e.g., jaw, tongue, lips, velum, etc.), together with the vocal-tract excitation (not shown

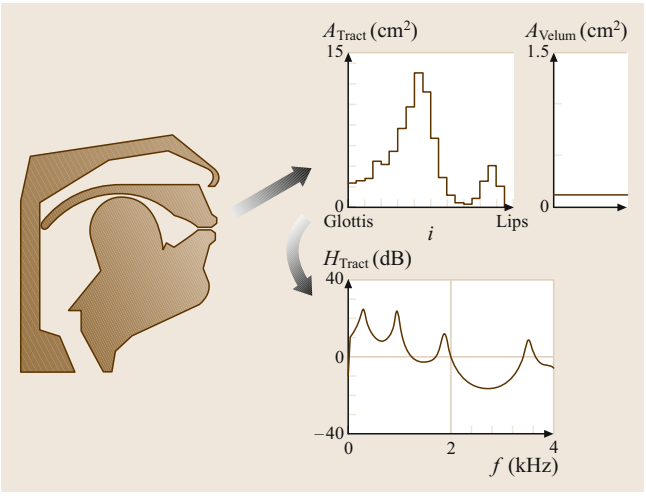


Fig. 19.4 The human speech-production system with glottis, vocal tract, and nasal tract

in Fig. 19.4; vibrations of the vocal folds of the glottis for voiced sounds, aspiration at the glottis and fricative noise generated at vocal tract constrictions for unvoiced sounds) allow for speech synthesis using articulatory parameters as controls. A complete articulatory synthesizer would use information such as the forces and timings of all relevant groups of articulatory muscles to drive a realistic mathematical model of glottis and vocal tract.

Clearly, any measurements of the vocal tract geometry are helpful for articulatory synthesis research. Such measurements are obtained by one of several techniques. For example, ultrasound pictures of the tongue shapes [19.11] can reveal important details of tongue movements. Nuclear magnetic resonance [19.12] is now fast enough to track vocal tract dynamics in real time.

From these kinds of data, so-called articulatory models (static models, e.g. [19.13], or dynamic models, e.g. [19.14]) then convert positional coordinates of the articulators to a series A_i^{TRACT}, $i = 1, \ldots, p$ of piecewise-constant vocal-tract cross-sectional areas that can be used to compute the vocal-tract acoustics. An example is shown in the top right of Fig. 19.4.

There is a multitude of ways to characterize the acoustics of the vocal tract. Here we will focus on a frequency-domain method (see [19.1] for others). Assuming plane wave propagation in the vocal tract (an assumption that is valid for frequencies under 4 kHz), the input/output characteristics of the vocal tract can be characterized by its chain matrix (also known as its $ABCD$ matrix) that links sound pressure at the lips P_{L} and volume flow at the lips $U_{\mathrm{L}}$to their counterparts at the glottis, P_{G} and U_{G}, respectively:

$$\begin{pmatrix} P_{\mathrm{L}} \\ U_{\mathrm{L}} \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} P_{\mathrm{G}} \\ U_{\mathrm{G}} \end{pmatrix} , \tag{19.1}$$

where the tract ABCD matrix is the product of individual section matrices as follows, each section with a different cross-sectional area A_i^{TRACT} and, optionally, length:

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} = \prod_{k=1}^{p} \begin{pmatrix} A_k & B_k \\ C_k & D_k \end{pmatrix} . \tag{19.2}$$

Here the ABCD matrix elements are

$$\begin{aligned} A_i &= \cosh(\Gamma_i \Delta) , & B_i &= -Z_i \sinh(\Gamma_i \Delta) \\ C_i &= -Z_i^{-1} \sinh(\Gamma_i \Delta) , & D_i &= \cosh(\Gamma_i \Delta) , \end{aligned} \tag{19.3}$$

with characteristic impedances Z_i, lossy propagation constants Γ_i, and vocal-tract section length Δ (assumed to be the same for all sections). Z_i and Γ_i depend on the cross-sectional area A_i^{TRACT}. Both of these quantities are determined by the (per-unit-length) inductance L_i, capacitance C_i, and the loss elements resistance R_i and conductance G_i [19.1, p. 34], and are given by:

$$Z_i = \sqrt{\frac{(R_i + \mathrm{i}\omega L_i)}{(G_i + \mathrm{i}\omega C_i)}} \tag{19.4}$$

and

$$\Gamma_i = \sqrt{(R_i + \mathrm{i}\omega L_i)(G_i + \mathrm{i}\omega C_i)} \tag{19.5}$$

with $\mathrm{i} = \sqrt{-1}$ and ω the radian frequency. There are several meaningful quantities that are easily computed from (19.1). One that we will need in Sect. 19.2 below is the *vocal-tract transfer function* that relates volume velocity at the lips to glottal volume velocity (glottal flow):

$$H = \frac{U_{\mathrm{L}}}{U_{\mathrm{G}}} = \frac{1}{A - C Z_{\mathrm{L}}} . \tag{19.6}$$

Here Z_{L} is the radiation impedance at the lips. From (19.6), we compute the sound pressure at the lips as $P_{\mathrm{L}} = H_{\mathrm{TRACT}} U_{\mathrm{G}}$, where $H_{\mathrm{TRACT}} = Z_{\mathrm{L}} H$. An example for H_{TRACT} is depicted in the lower right of Fig. 19.4.

The inverse Fourier transform of H_{TRACT} is the impulse response of the vocal tract that, convolved with the actual glottal flow, results in synthetic speech. Many more details including information on how to model the glottal and noise excitations can be found, for example, in [19.15].

In general, articulatory synthesis produces intelligible synthetic speech, but today its output is still far from natural sounding. The reason for this fact is that each of the several models that are employed in the process has to be extremely accurate in reproducing the characteristics of a given *single* speaker. Also note that most of these models largely depend on expert guesses (*rules*) and not enough on observed data. While the promise of high-quality articulatory synthesis is the possibility of easily modifying any model with the objective of changing underlying speaker characteristics, the complexities and computational requirements of this method still hamper its broad adaptation in practical speech synthesizers.

Formant Synthesis

Formant synthesis encapsulates the vocal tract as a black box, aiming to reproduce its input/output characteristics, namely (19.6), in form of formant filters. Further approximations are made for the radiation impedance Z_{L} and for the excitation mechanisms. More generally, formant synthesizers instantiate the so-called source–filter theory of speech production in a terminal analog fashion. The theory postulates that there is no influence of the

filter back on the source (contrary to articulatory synthesizers). Instead of vocal tract areas, formant frequencies and bandwidths are used as control factors. The filters are excited either with a pulse train or with noise.

Assuming $Z_L = 0$ in (19.6) and neglecting any losses in the vocal tract, the transfer function (19.6) simplifies to $H = A^{-1}$. For the i-th tract section, the A element of its chain matrix becomes (19.3):

$$A_i = \cos\left(\frac{\omega\Delta}{c}\right) \quad \forall i = 1, \ldots, p . \tag{19.7}$$

Here c is the speed of sound. Representing the vocal tract using p sections, each of length Δ, the round-trip wave propagation delay through a section is $2\Delta/c$. This delay corresponds to the delay operator z^{-1} in a digital implementation. Consequently, (19.7) may be written as

$$\cos\left(\frac{\omega\Delta}{c}\right) = \frac{z^{-1/2} + z^{+1/2}}{2} = \frac{z^{1/2}}{2}(1 + z^{-1}) . \tag{19.8}$$

By chaining the individual terms for the entire vocal tract, we get a polynomial of order p, of the form

$$\frac{U_L}{U_G} = \frac{z^{-p/2}}{\sum_{i=0}^{p} a_i z^{-i}} \Rightarrow \sum_{i=0}^{p} a_i z^{-i} U_L = z^{-p/2} U_G . \tag{19.9}$$

In the time domain, the right-hand form of (19.9) expands to

$$\sum_{i=0}^{p} a_i u_L(n-i) = u_G\left(n - \frac{p}{2}\right) . \tag{19.10}$$

Making the simple substitutions $s(n) \equiv u_G$ and $u_G(n - p/2) \equiv e(n)$, we arrive at the standard linear prediction coding (LPC) equation [19.16]:

$$s(n) = e(n) - \sum_{i=1}^{p} a_i s(n-i) \quad \text{with } a_0 = 1 \tag{19.11}$$

showing that we can represent a lossless non-nasal vocal tract using a standard LPC polynomial.

Serial Formant Synthesizer. Assuming p even and only conjugate complex roots, the LPC polynomial (19.9) can be factored into its roots, resulting in

$$\begin{aligned}\frac{U_L}{U_G} &= \prod_{i=1}^{p/2} \frac{(1 - z_i)(1 - z_i^*)}{(1 - z_i z^{-1})(1 - z_i^* z^{-1})} \\ &= \prod_{i=1}^{p/2} \frac{1 - 2\mathrm{e}^{-\sigma_i T}\cos(\omega_i T) + \mathrm{e}^{-2\sigma_i T}}{1 - 2\mathrm{e}^{-\sigma_i T}\cos(\omega_i T) z^{-1} + \mathrm{e}^{-2\sigma_i T} z^{-2}} ,\end{aligned} \tag{19.12}$$

where the i-th root $z_i = \mathrm{e}^{-\sigma_i T}\mathrm{e}^{-\mathrm{i}\omega_i T}$, the asterisk (*) denotes the conjugate complex, and T is the sampling interval. Note that the frequency-independent numerator of each factor (i. e., the filter element) normalizes the gain of that factor to be unity at zero frequency ($z = 1$).

Figure 19.5 shows the frequency characteristic of a single filter element. Note that the peak gain is given by its quality factor $Q_i = \omega_i/2|\sigma_i|$, and the bandwidth of the filter in Hertz is $B_i = F_i/Q_i = \omega_i/2\pi Q_i = |\sigma_i|/\pi$. Cascading $p/2$ second-order filter sections instantiates (19.12). Note that specifying the formant frequencies F_i and bandwidths B_i fully determines the transfer function of the chained filters.

Parallel Formant Synthesizer. A serial synthesizer is a good approach for open, non-nasal vocal tract configurations (e.g., for vowels and liquids). In order to reproduce the speech spectra of obstruents and nasals, however, we need to control individual filter section amplitudes in addition to formant frequencies and bandwidths. This is possible if individual filter sections are connected in parallel, each with its own gain factor. The price we pay for the additional flexibility in matching spectrum levels is that spurious zeros are introduced between the formant frequencies.

In order to arrive at a parallel filter representation, we start from (19.12) and expand the product form into a summation form through partial fractions:

$$\begin{aligned}\frac{U_L}{U_G} &= \prod_{i=1}^{p/2} \frac{1 - 2\mathrm{e}^{-\sigma_i T}\cos(\omega_i T) + \mathrm{e}^{-2\sigma_i T}}{1 - 2\mathrm{e}^{-\sigma_i T}\cos(\omega_i T) z^{-1} + \mathrm{e}^{-2\sigma_i T} z^{-2}} , \\ &= \sum_{i=1}^{p/2} \frac{A_i - B_i z^{-1}}{1 - 2\mathrm{e}^{-\sigma_i T}\cos(\omega_i T) z^{-1} + \mathrm{e}^{-2\sigma_i T} z^{-2}} , \\ &\approx \sum_{i=1}^{p/2} \frac{A_i}{1 - 2\mathrm{e}^{-\sigma_i T}\cos(\omega_i T) z^{-1} + \mathrm{e}^{-2\sigma_i T} z^{-2}} .\end{aligned} \tag{19.13}$$

Here, the residual terms A_i (the *gain*) and B_i (the phasing of the resonance, neglected in most practical realizations) allow for the additional control that we desire.

Combined Serial/Parallel Formant Synthesis. Including a serial filter realization for vowels and liquids and a parallel filter realization for nasals and fricatives is clearly advantageous. Figure 19.6 shows a block diagram of Klatt's synthesizer [19.17]. Also shown is a parametric voicing source that can be tuned for dif-

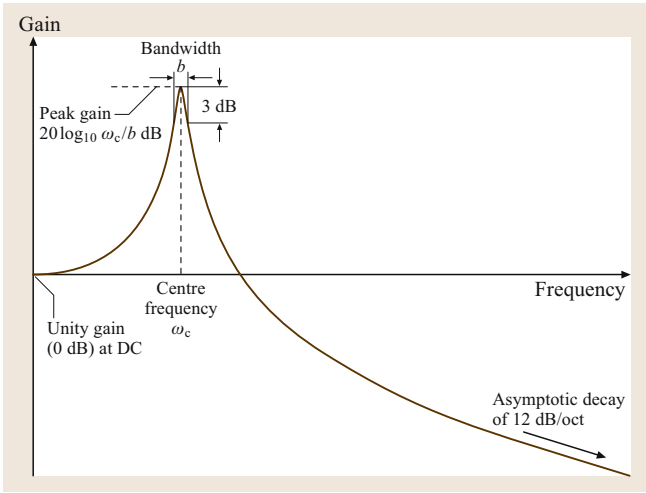


Fig. 19.5 The frequency response of a single second-order filter section of a formant synthesizer with angular center frequency ω_c and bandwidth $b = 2\pi B$

ferent voices or speaking styles. The synthesizer also includes an aspiration/breathiness component added to the glottal source that assists in the production of voiced fricatives. A commercial system that makes use of articulatory and acoustic-phonetic knowledge and drives the Klatt synthesizer is HLSyn [19.18]. Readers interested in rule-based and formant synthesis can find more information in Chap. 20.

Formant synthesis is the preferred method for creating speech stimuli for research in speech perception, given the high level of control such experiments require. Another advantage of formant synthesizers is their moderate computational requirements. For example, memory requirements can be as little as 1 MB. Therefore, formant synthesis is ideal for applications such as in handheld devices, for example, talking dictionaries, calendars, etc., or even in cell phones (e.g., for reading back names and key presses). Intelligibility is high, but naturalness is generally lacking. Matching a target speaker is usually impossible.

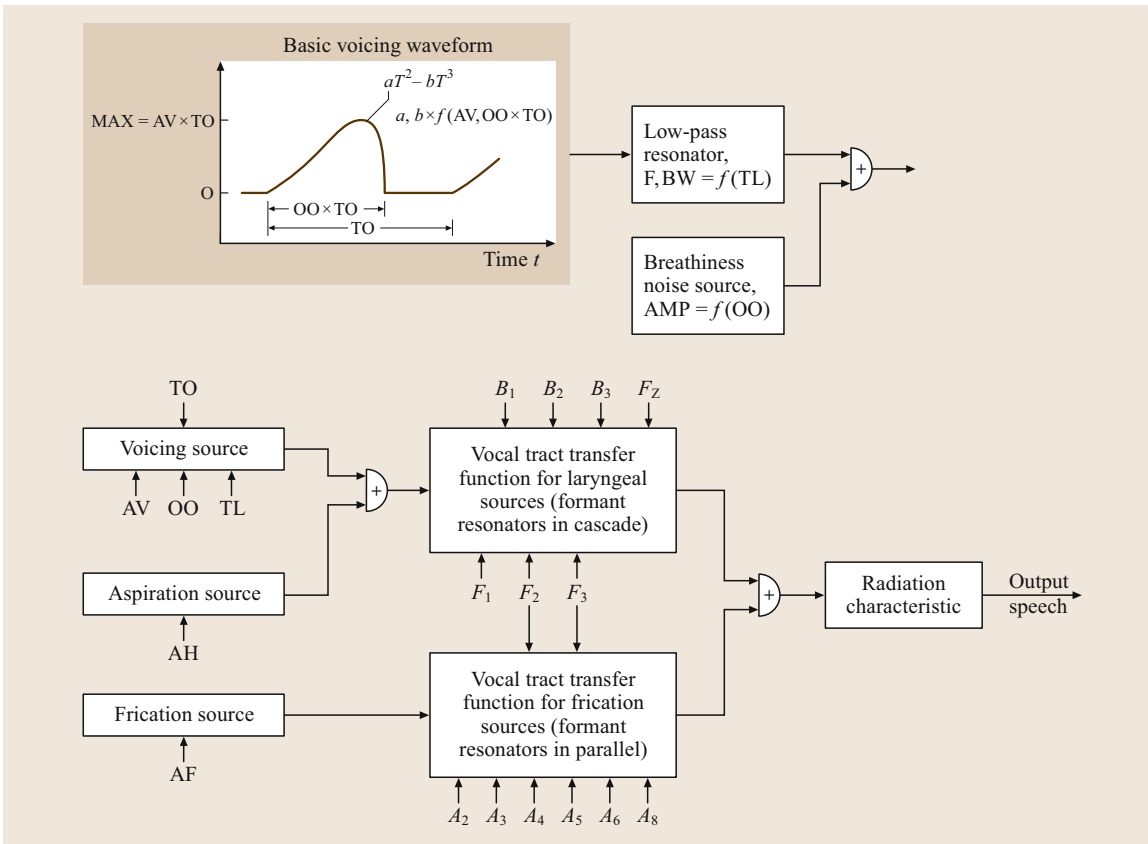


Fig. 19.6 Hybrid serial/parallel synthesizer of Klatt

Concatenative Synthesis

Concatenative synthesis employs segments of recorded speech from a prerecorded inventory (voice database). A block diagram of a typical concatenative TTS system is shown in Fig. 19.7. The phonetic and prosody targets provided by the frontend on the left act as a *fuzzy* query into the voice database that stores all sound units. The sequence of units that optimally match the input query for a to-be-synthesized sentence are retrieved, assembled, and handed off to the speech waveform modification and synthesis module on the right. Unit assembly and waveform synthesis are both backend tasks. Note that changing the language requires changing the frontend, but usually also requires changing the store of sound units. Changing the target speaker (e.g., from male to female) also usually requires changing the store of sound units.

The sound store may contain one or more of different types of speech units. For example, using phones (e.g., about 50 for English) leads to unsatisfactory joints between units because of the large coarticulatory effects between adjacent phones. Intuitively, longer units are more likely to result in higher-quality synthesis, given that the rate of concatenations (how many unit-to-unit transitions occur per second of speech) is lower than in the case of shorter units. For general TTS applications, however, longer units are impractical because of the tremendous multiplicity of possible unit variants [19.19]. Therefore, using whole word units is impossible because of the huge demands on a voice talent that would have to read a few hundreds of thousands of words in a consistent voice and manner but also in different prosodic contexts. Given these contradictory requirements, most TTS implementations until the mid-1990s used one of two types of inventory units: the diphone or the demisyllable.

A diphone is the segment of speech that starts at the middle of one phone and extends to the middle of the next phone. It brackets exactly one phone-to-phone transition. The cut points at the middle of the phones are in the acoustically most stable region. Note that the average length of a diphone is identical to that of a phone. Because each diphone has *two* phone identities, the theoretical size of a diphone inventory is quadratic in the number of the phones a language has. However, because of the fact that a language might not use all of these combinations, the actual number of diphones that needs to be in the inventory is significantly smaller. For example, English requires an inventory of at least 1000 diphones [19.20].

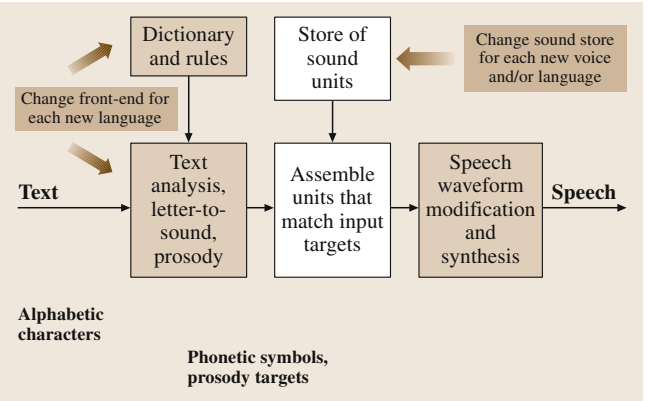


Fig. 19.7 Block diagram of a concatenative synthesizer

Demisyllables are alternative units for concatenative synthesis [19.21]. A demisyllable encompasses half a syllable; that is, either the syllable-initial portion up to the first half of the syllable nucleus, or the syllable-final portion starting from the second half of the syllable nucleus. The number of demisyllables in English is roughly the same as the number of diphones. Because demisyllable units are usually longer than diphones and allow for better capture of longer-term coarticulation effects compared to diphones, they should pose fewer concatenation problems.

With a minimum-size inventory, units must be modified by signal processing to match the frontend targets and to smooth over the concatenation points. After recording the voice talent, special care must be spent on selecting the most appropriate instances of each unit to be included in the inventory. Semi-automatic tools can help in finding those instances of units that minimize concatenation problems when used in later synthesis [19.22, pp. 199-222].

The offline (not carried out at synthesis time) process of selecting optimum units can be improved further by incorporating it in the online synthesis process. The quality of the synthesized speech should increase given that full context information is available at synthesis time, but only general statistical information exists when done offline. This idea originated at advanced telecommunications research (ATR) in Japan in the late 1980s [19.23]. The resulting new subcategory of concatenative synthesis is now called *unit-selection synthesis* [19.24]. It was largely enabled by the ever-increasing power of computers. Unit-selection synthesis adds the problem of how to do fast searches of potentially millions of options to the set of tasks in speech synthesis.

Unit-Selection Synthesis. Different from earlier concatenative synthesizers, unit-selection synthesis automatically picks the optimal synthesis units (on the fly) from an inventory that can contain thousands of examples of a specific unit (such as a diphone), and concatenates the selected sequence of units to produce the synthetic speech. This process is outlined in Fig. 19.8, which shows how the method must dynamically find the best path through the unit-selection network corresponding to the sounds for the word 'two'. The optimal selection of the sequence of units depends on factors such as spectral similarity at unit boundaries and on matching prosodic targets set by the front-end. In the following, we will highlight the essential steps, as used in the CHATR speech synthesis system [19.25]. Many more aspects and details are covered in Chap. 21.

There are two principal kinds of cost measures that guide unit selection. One is unit segmental distortion (USD); another name for it is *target cost*. USD captures the spectral differences between candidate units in the inventory and a hypothetical target, noting that the real target is not available at synthesis time. Therefore, USDs have to be estimated from available features for the target, given the sentence to be synthesized. An example would be to represent the cost of selecting an /ah/ from the word *want* to synthesize the word *cart*. The USD in this case is the penalty for violating the desired /ah-r/ context by selecting /ah/ in an /ah-n/ context. USDs are represented by the nodes in Fig. 19.8.

The other kind of cost is unit concatenative distortion (UCD), also called *join cost*. It measures the spectral discontinuity across the boundaries of candidate units. Since the candidates exist in the inventory, UCDs can be computed directly from the candidates. UCDs are represented by the arcs (arrows) in Fig. 19.8.

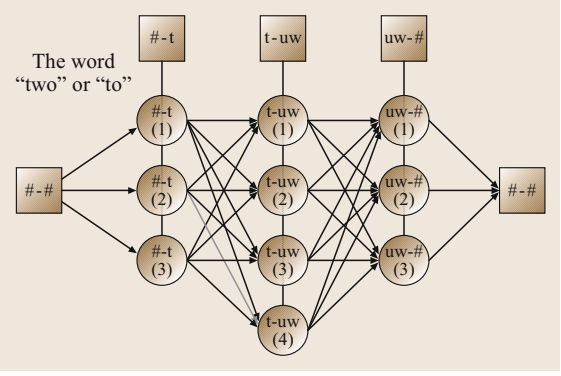


Fig. 19.8 Viterbi search of a diphone inventory for the words 'two' or 'to'

We formalize the unit selection process as follows. We start with denoting t_j to be the j-th (idealized) target unit in the string $T = \{t_1, \ldots, t_j, \ldots, t_N\}$ that spans the sentence to be synthesized. Similarly, we form a set of N corresponding units selected from the inventory to be $\Theta = \{\theta_1, \ldots, \theta_j, \theta_{j+1}, \ldots, \theta_N\}$. Then the total unit selection cost for any path through the network depicted in Fig. 19.8 is given by

$$d(\Theta, T) = \sum_{j=1}^{N} d_{\mathrm{u}}(\theta_j, t_j) + \sum_{j=1}^{N-1} d_{\mathrm{t}}(\theta_j, \theta_{j+1}) \quad (19.14)$$

where $d_{\mathrm{u}}(\theta_j, t_j)$ is the USD or target cost between target unit t_j and candidate unit θ_j and $d_{\mathrm{t}}(\theta_j, \theta_{j+1})$ denotes the UCD or join cost between the two neighboring units θ_j and θ_{j+1}. The optimal path (the optimal sequence of units) can be found efficiently using the Viterbi method that scales linearly with the grid size (the number of time steps multiplied by the number of candidate units in a column of the network) [19.26].

The join cost has to be zero if both units are natural neighbors in the inventory, that is, if they were recorded in sequence (e.g., came from the same word) because, by definition, there is no spectral mismatch in this case. The exact type of join costs to use is an area of active research that tries to gauge a listener's perceptual *annoyance* of spectral mismatches [19.27].

USD or target costs are conceptually more difficult to understand. In practice, they are also more difficult to instantiate. Formally, we start with

$$d_{\mathrm{u}}(\theta_j, t_j) = \sum_{i=1}^{q} w_i^{\mathrm{t}} \phi_i[\mathrm{T}_i(f_i^{\theta_j}), \mathrm{T}_i(f_i^{t_j})] \,, \quad (19.15)$$

where the w_i^{t} are trained weights (see later), and the $\mathrm{T}_i(\cdot)$ can either be continuous functions for a set of features f_i such as segmental pitch, power, or duration, or a set of integers in case of categorical features f_i such as unit identity, phonetic class, etc. In the latter case, ϕ_i is looked up in a distance table. Otherwise, a simple quadratic distance may suffice:

$$\phi_i[\mathrm{T}_i(f_i^{\theta_j}), \mathrm{T}_i(f_i^{t_j})] = [\mathrm{T}_i(f_i^{\theta_j}) - \mathrm{T}_i(f_i^{t_j})]^2 \,. \quad (19.16)$$

The training of the weights w_i^{t} is done offline. For each phoneme in each phonetic class in the training speech database (which might be the whole recorded inventory), we treat each exemplar of a unit as a target and all others as candidate units. Let L be the number of phones in the phone set and M the number of lowest target-cost

candidates that we wish to consider. Then, a $(L \times M) \times q$ rectangular matrix $\boldsymbol{\Phi}$ for this set is given by

$$\boldsymbol{\Phi}_{(L\times M)\times q} = [\phi_1, \ldots, \phi_{\mathrm{L}}]^{\mathrm{T}} , \tag{19.17}$$

where T denotes matrix transposition. Similarly,

$$\boldsymbol{w}^{\mathrm{t}}_{qx1} = [w^{\mathrm{t}}_1, \ldots, w^{\mathrm{t}}_q]^{\mathrm{T}} \tag{19.18}$$

is the desired vector of weights. Furthermore,

$$\boldsymbol{a}_{(L\times M)\times 1} = [d_{\mathrm{a}}(\theta_1, t_1) \ldots d_{\mathrm{a}}(\theta_M, t_1)| \ldots |d_{\mathrm{a}}(\theta_1, t_L) \ldots d_{\mathrm{a}}(\theta_M, t_L)]^{\mathrm{T}} \tag{19.19}$$

is a vector of acoustic distances d_{a}, similar to the UCD join cost, but now applied over all corresponding frames of a pair of candidates and targets units using, for example, dynamic time warping (DTW). The optimal weight set $\boldsymbol{w}^{\mathrm{t}}$ then solves the least-squares system of linear equations

$$\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\boldsymbol{w}^{\mathrm{t}} = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{a} . \tag{19.20}$$

Because of coverage issues, unit selection appears to be well suited for limited-domain applications such as synthesizing telephone numbers to be embedded within a fixed carrier sentence. Even for more open-domain applications, such as email reading, advanced unit selection can reduce the number of unit-to-unit transitions per sentence synthesized and, consequently, increase the segmental quality of the synthetic output. Furthermore, the use of multiple instantiations of a unit in the inventory, taken from different linguistic and prosodic contexts, reduces the need for prosody modifications that potentially degrade naturalness. Consequently, today's most natural sounding unit-selection systems employ inventories that comprise many dozens of hours of speech, in particular when more than one speaking style (e.g., newsreader) or emotion (e.g., neutral) is to be covered [19.28] (Chap. 25).

HMM-Based Synthesis. An alternative approach to instantiating the USD/target costs in unit selection is based on clustering contexts offline and selecting the best unit at runtime from the optimal cluster, given the context information for the sentence to be synthesized [19.29]. This task is well known in acoustic modeling in speech recognition. Therefore, a similar approach using hidden Markov models (HMMs) can also be used for this step in synthesis. Using HMM states as units and adding a speech generation method from HMM state data (spectra and excitation) leads to an HMM-based synthesis method [19.30].

In the next section, we will look into some of the signal representations typically used in concatenative TTS systems. The idea is that limited signal processing may alleviate the combinatorial problems [19.19] that potentially degrade even the intelligibility of a TTS system because of the lack of appropriate units in the inventory. Signal processing may also allow speaker transformations, thus lowering the cost of generating new voice inventories. We will look into speaker and speech transformations in Sect. 19.3.

19.2 Speech Representations and Signal Processing for Concatenative Synthesis

A speech representation suitable for concatenative synthesis satisfies the following requirements.

1. In order to minimize disk and/or memory requirements, voice inventories are stored in compressed form at high quality. The encoding/decoding process should be of low computational complexity.
2. Encoding/decoding has to be perceptually transparent for maximum naturalness. Any perceptually detectable distortions should be avoided. Early examples of concatenative synthesizers violated this requirement, sacrificing naturalness while maximizing intelligibility.
3. Algorithms used for encoding and decoding need to allow for random access to individual units of speech (e.g., diphones).
4. Ideally, the selected speech representation must allow for natural-sounding modifications of prosody. Note, however, that for most signal processing algorithms, modifying pitch more than a few percent may destroy naturalness.
5. Advanced signal processing might avoid extensive and impractically long recording sessions. Advanced voice conversion might be used to approximate emotional speaking styles [19.28] (Chap. 25). However, many people believe that today's voice conversion algorithms still do not produce the necessary quality.

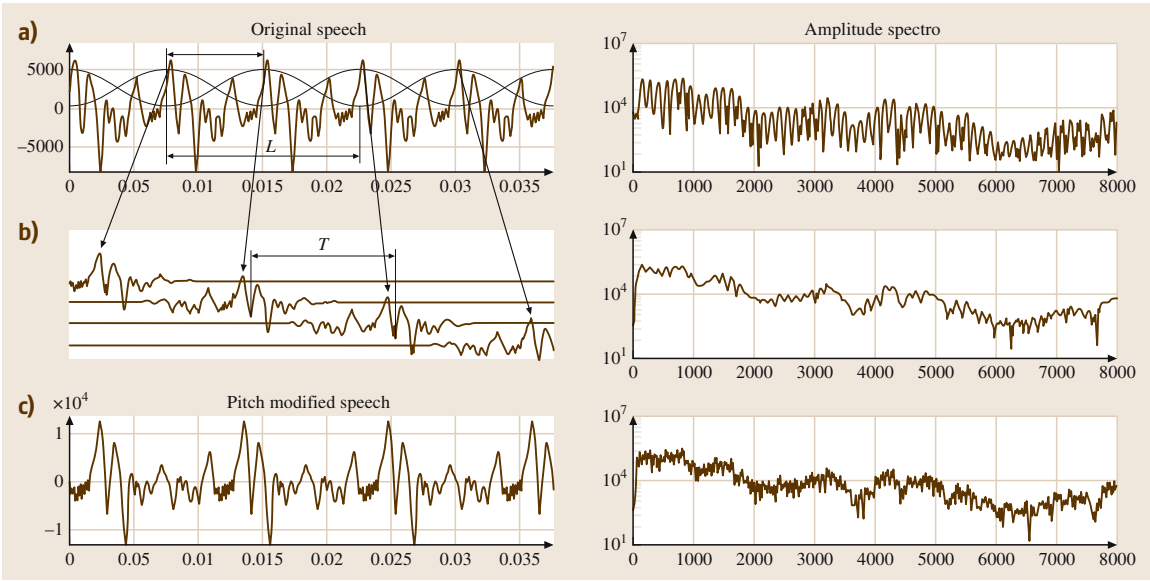


Fig. 19.9a–c Time-domain pitch synchronous overlap add (TD-PSOLA, after [19.31, Fig. 10.1, p. 252])

Given this list of requirements, in the following, we will now outline three classes of speech signal processing algorithms with their (native) speech representations: low-complexity time-domain algorithms such as TD-PSOLA (using the waveform of the speech signal directly), LPC-based representations, and frequency domain-based speech representations. Common to all of them is the fact that changes in segment durations are almost trivially tied to modifications of parameter update rates. They are different, however, in the manner in which they facilitate pitch changes and allow or disallow changes in spectral envelopes and spectral fine structure of the speech signal.

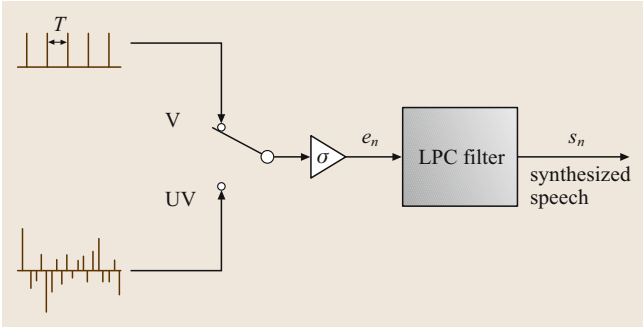


Fig. 19.10 Block diagram of LPC synthesis. Voiced excitation is facilitated by periodic pulses with pitch period T. Gaussian noise is used for unvoiced excitation. The excitation is amplified by a gain factor σ

19.2.1 Time-Domain Pitch Synchronous Overlap Add (TD-PSOLA)

In its simplest form, time-domain pitch-synchronous overlap add (TD-PSOLA; see, e.g., [19.32]) consists of extracting two pitch periods from a voiced speech signal, windowing each segment with a Hanning window centered on one glottal closure (maximum excitation) point. This positioning of the window weighs down the signal in the vicinity of the previous and next glottal closure events as shown in Fig. 19.9. Note that the figure depicts time-domain signals on the left and spectra on the right. Panel (a) shows the original signal. The extracted and windowed signal traces (b) are recombined in (c), after shortening (for increased pitch) or padding with zero amplitude signal samples (for lowering pitch). The resulting reconstructed signal of a different pitch value is shown in panel (c). A specific variant of TD-PSOLA is to apply the method on the LPC filter excitation (pitch-synchronous residual excited linear prediction (PSRELP), see [19.33]) instead of on the speech waveform. As an option, PSRELP allows for smoothing of the spectral envelope at concatenation points by modifying the LPC filter, alleviating concatenation problems somewhat. Both methods, however, can

lead to audible glitches since there is no obvious way of smoothing the waveforms that TD-PSOLA is applied to. While the PSRELP variant allows for changing the spectral envelope, the spectral fine structure cannot be controlled by either method.

19.2.2 LPC-Based Synthesis

We introduced linear prediction (LPC) in (19.11) in Sect. 19.1.2, as a stepping stone towards formant synthesis. In the simplest version of LPC synthesis, depicted in Fig. 19.10, the excitation signal $e(n)$ is either a train of pulses (for voiced speech) or white Gaussian noise (for unvoiced speech). The LPC filter gives the synthetic speech the desired spectral envelope, matching the formants without explicit formant identification. This is enough to create intelligible speech but fails to produce natural-sounding speech because of the simplistic excitation model. This kind of LPC synthesis clearly is not natural sounding. However, because of its relationship to vocal tract geometry, LPC lends itself to spectral modifications that are based on geometric differences of vocal tracts (e.g., male-to-female speech conversion). In addition, note that some TTS systems employ advanced LPC-based encoders and decoders such as code-excited linear prediction (CELP) or multipulse [19.34], and/or coders that employ glottal model excitation pulses. While using CELP or multipulse excitation raises the quality dramatically, only glottal models have the advantage of meaningful interpretation in terms of human speech production and, consequently, lend themselves to sensible smoothing and interpolation. It is difficult, however, to match an individual speaker's voice characteristic with a glottal model-based synthesizer. Both methods allow for changes in spectral envelope, but fall somewhat short when we would like fine control of the spectral details.

19.2.3 Sinusoidal Synthesis

The main disadvantage of the LPC synthesis of Fig. 19.10 is its lack of faithful reproduction of a speaker's voice characteristics and the lack of fine control of the speech spectrum. It can be shown that the amplitude and phase relationships of the first few harmonics contain crucial information on speaker identity (independent of the super-segmental prosodic and phonetic cues). Therefore, modeling speech harmonics directly using a sinusoidal speech representation seems to be a more appropriate approach towards meeting the transparency requirement. As with TD-PSOLA, in sinusoidal synthesis we have the choice of including LPC filtering to allow for smoothing at concatenation points and for other modifications of the spectral envelope. In contrast to TD-PSOLA, here pitch modifications are done in the frequency domain by stretching or compressing the spectrum.

Let $s(t)$ be a signal of interest, either speech or LPC excitation, then [19.35]

$$\begin{aligned} s(t) &= \sum_{l=1}^{L(t)} A_l(t)\cos[\boldsymbol{\Omega}_l(t)]\ , \text{ with } \boldsymbol{\Omega}_l(t) \\ &= \int_{t_l}^{t} \omega_l(\sigma)\,\mathrm{d}\sigma + \phi_l\ . \end{aligned} \tag{19.21}$$

Here $A_l(t)$ is the slowly time-varying amplitude of the l-th sine wave, $\boldsymbol{\Omega}_l(t)$ its instantaneous phase, ω_l the radian frequency of the l-th sine wave, and ϕ_l its phase. During analysis of a speech signal, we have to estimate the harmonic frequencies and phases. Potential problems are the *birth* and *death* of specific sinusoids and the representation of unvoiced speech.

Several variants of the sinusoidal approach exist. One is the harmonic-plus-noise model representation (HNM) [19.36]. It splits the speech spectrum into a low-frequency voiced part that is modeled by the sinusoidal approach, and a high-frequency unvoiced part that represents the noise-like characteristic of speech spectra at high frequencies. Another approach is Straight [19.37], which focuses on highly accurate estimation of the harmonic amplitudes.

19.3 Speech Signal Transformation Principles

The goal of speech signal transformation is to create output speech with characteristics that are different from the speech from which it was derived. This may include changing speaker identity (making the output speech sound like a different speaker), or may convert speech recorded in a neutral newsreader style so that it sounds angry or happy (change its emotional tone). Therefore, one important reason to transform speech is to regenerate its variability that limited recordings cannot capture [19.28]. There are many kinds of variability

in speech, including speaker variability, linguistic inter and intra-speaker variability, and task variability. The acoustic correlates of these dimensions are variations in pitch, duration, amplitude, and spectrum. We already discussed how to change these low-level attributes, with the exception of spectral characteristics, which we will cover in this section.

We will discuss voice transformation to change speaker identity, voice alteration targeted at changing gender, individual differences, or even dialect, and modifications to speaking style and emotions for a given speaker. Although the methods outlined below may serve all of these applications, there are important differences between them. While changing the speaker identity can multiply the return on the effort that went into creating a single high-quality TTS voice by enabling offering several derived voices instead of just the one originally recorded, modifying intra-speaker characteristics such as speaking style or emotion avoids recording voice databases in that specific style or emotion.

What are the speech and speaker characteristics that might be changed by signal processing? One component of linguistic variability of speech is caused by variations in intonation. In addition, phonemes exhibit different spectral properties depending on context (coarticulation), speaking style/emotion, and speaker identity. Average pitch and formant frequencies are higher for females than for males. Different dialects cause the use of different phonemes for a word and/or the use of different allophones for the same phoneme in a given context. In the following, we will highlight methods for modifying spectral characteristics. However, we will also look at *individualized* prosody models that capture a specific speaker and/or emotional/speaking style. Details on more-expressive speech synthesis can be found in Chap. 25.

19.3.1 Prosody Transformation Principles

Changing segmental signal characteristics such as duration, amplitude, and pitch enables us to change supersegmental aspects of prosody as well with the goal of conveying a different emotion or a different speech act. This assumes, however, that we know what longer-term changes are required. Again, a data-driven approach seems most promising for capturing the way a specific speaker uses prosody in a specific context (e.g., in a dialog scenario) and for a specific speech act (e.g., a greeting). Prosody models trained on this information then allow a TTS system to request the right prosody for the given context. For a unit-selection synthesizer, this allows the unit selection module to do a good job of retrieving the correct speech segments that might minimize prosodic changes that might be necessary.

Any of the parametric intonation/prosody models summarized previously in Sect. 19.1.1 and detailed in Chap. 23 may be included in the signal transformation framework introduced below, provided that it includes model parameters of the chosen prosody model or representation.

19.3.2 Principle Methods for Changing Speaker Characteristics and Speaking Style

Here we discuss ways to transform the speech of one speaker to sound like speech from another speaker. The same methods can be used to change the speaking style or emotional tone for a given speaker. The simplest approach for mapping source speech (what we have) to target speech (what we intend to make it sound like) with another set of characteristics is to use two codebooks of vector-quantized parameter vectors, each trained on one of the two sets. Then we train a mapping function between corresponding codebook entries using a parallel speech corpus where both speakers uttered the same sentences. This technique was pioneered by Abe at Nippon Telephone & Telegraph (NTT) in Japan [19.38]. Time alignment of vectors from one set with the corresponding vectors from the other is not trivial because of differences in vowel reductions, phone omissions or additions, or other individual pronunciation differences. Dynamic time warping (DTW) may be used to establish this correspondence and histograms of these correspondences serve as weights for the mapping function:

$$\boldsymbol{S}_i^{(A\to B)} = \frac{\sum_{j=1}^{J} w_{ij}\boldsymbol{S}_j^{(B)}}{\sum_{j=1}^{J} w_{ij}} \,. \tag{19.22}$$

Here A and B represent source and target sets, respectively, $\boldsymbol{S}_i^{(A\to B)}$ is the i-th speech vector of the mapped source codebook, and w_{ij} are the histogram counts of how often vector i of codebook A was found to correspond to vector j in codebook B, itself containing J vectors $\boldsymbol{S}_j^{(B)}$. We can interpret (19.22) as an interpolation between different target spectral vectors. Therefore, it is important to choose a representation $\boldsymbol{S}$ where linear interpolation makes sense and does not lead to artifacts.

An improved method of mapping features from a source corpus to those of a target corpus is to esti-

mate an appropriate transformation function. The most popular method for doing this is based on Gaussian mixture models (GMMs). One of the early adopters of GMM-based methods for voice conversion was Stylianou [19.39] (Chap. 24). Let ω_m denote the m-th acoustic class out of a total of M classes. In a Gaussian mixture density model, the probability density functions of any of the K observation vectors $\boldsymbol{x}_k (k = 1, \ldots, K)$ is given by:

$$p(\boldsymbol{x}_k) = \sum_{m=1}^{M} P(\omega_m) p(\boldsymbol{x}_k|\omega_m) = \sum_{m=1}^{M} c_m N(\boldsymbol{x}_k, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m) , \quad (19.23)$$

where $\boldsymbol{x}_k$ is an L-dimensional random vector of speech features. The conditional probability density for each of the M classes ω_m is assumed to be a Gaussian component density $N(\boldsymbol{x}_k, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$, and the c_m (the probabilities for classes ω_m) are the related mixture weights normalized to have unity sum. Each L-variate Gaussian component density is of the form:

$$N(\boldsymbol{x}_k, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m) = \frac{1}{(2\pi)^{L/2} |\boldsymbol{\Sigma}_m|^{1/2}} \times \exp\left[-\frac{1}{2} (\boldsymbol{x}_k - \boldsymbol{\mu}_m)' \boldsymbol{\Sigma}_m^{-1} (\boldsymbol{x}_k - \boldsymbol{\mu}_m) \right] . \quad (19.24)$$

The L-dimensional mean vector $\boldsymbol{\mu}_m$ and the covariance matrix $\boldsymbol{\Sigma}_m$ determine the center location and the spread, respectively, of the m-th mixture. Equation (19.24) expresses arbitrary distributions as a linear combination of Gaussians. If we associate the m-th component with a specific acoustic class (e.g., liquids, fricatives), it should be intuitively clear that mapping techniques based on (19.23) tend to be more robust than those based on (19.22), because the simpler approach does not distinguish between different acoustic classes. In addition, the simpler codebook mapping approach lacks a mechanism for generalizing to unseen data that the Gaussian representation provides.

How do we determine the mapping function $\boldsymbol{F}_x(\boldsymbol{x})$ that converts estimates of the source speech $\boldsymbol{x}_k$ into estimates of the target speech $\tilde{\boldsymbol{y}}_k$? We make the sensible assumption that this function should minimize the mean-squared error between a transformed set $\tilde{\boldsymbol{Y}} = \left\{ \tilde{\boldsymbol{y}}_k = \boldsymbol{F}_x(\boldsymbol{x}_k), k = 1, \ldots, K \right\}$ of time-aligned source data and the actual training set $\boldsymbol{Y} = \{\boldsymbol{y}_k, k = 1, \ldots, K\}$ of target data. To start, let $h_m(\boldsymbol{x}) = P(\omega_m|\boldsymbol{x})$ be the conditional probability of $\boldsymbol{x}$ belonging to class ω_m. With (19.23) and Bayes' rule, we find

$$h_m(\boldsymbol{x}) = P(\omega_m|\boldsymbol{x}) = \frac{P(\omega_m) p(\boldsymbol{x}|\omega_m)}{p(\boldsymbol{x})} = \frac{c_m N(\boldsymbol{x}, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)}{\sum_{i=1}^{M} c_i N(\boldsymbol{x}, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)} . \quad (19.25)$$

Then the desired mapping function $\boldsymbol{F}_x(\boldsymbol{x})$ is chosen to be of the form:

$$\boldsymbol{F}_x(\boldsymbol{x}) = \sum_{m=1}^{M} h_m(\boldsymbol{x}) \left[\boldsymbol{v}_m + \boldsymbol{\Gamma}_m \boldsymbol{\Sigma}_m^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_m) \right] , \quad (19.26)$$

where the unknown L-dimensional vectors $\boldsymbol{v}_m$ and the unknown $L \times L$ matrices $\boldsymbol{\Gamma}_m$ are determined by solving normal equations for the least-squares problem between $\boldsymbol{Y}$ and $\tilde{\boldsymbol{Y}}$. Note that this method may be improved further by basing the conversion function on joint source and target densities [19.40] instead of basing them solely on source densities as highlighted here. More on voice transformation can be found in Chap. 24 by Stylianou.

19.4 Speech Synthesis Evaluation

Evaluation of TTS systems is critical for many reasons. One reason is to gauge the progress in the field; another is to pick the best system for a specific application. Because so many specific interests might come to bear, it is impossible to list recipes for proper evaluation procedures that would fit any scenario. Short of that, we can only give broad guidelines here.

There are three quality criteria that one might be interested in: accuracy, intelligibility, and naturalness. They might overlap with each other with respect to which part of a TTS system impacts on which criteria. We define the accuracy of a TTS system as the ability to read a given input text the way a knowledgeable human reader would. For obvious reasons, accuracy is the one criterion solely homed in the TTS frontend. Contrary to this, unsatisfactory intelligibility or naturalness is much more difficult to tie to any single component of a TTS system. Between the latter two, designers of formant synthesizers primarily aim at maximizing intelligibility because higher naturalness can be achieved with unit-

selection methods at the cost of higher computational complexity. Vice versa, some of the newer concatenative synthesis systems may overemphasize naturalness with the result of lower intelligibility.

Accuracy may be evaluated by running a text corpus of task-relevant acronyms and abbreviations within appropriate carrier sentences through the TTS frontend and judging the generated output *text*. Moreover, evaluating intelligibility and naturalness requires conducting elaborate listening tests. In intelligibility tests, long known from testing speech coders, word or sentence lists are presented and subjects write down the words they hear [19.41]. However, evaluating the intelligibility of a TTS system clearly exposes more issues and aspects than are covered by standard intelligibility tests that rely on word lists. On a sentence level, for example, incorrect prosody can destroy intelligibility/comprehension. So far, a generally accepted standard intelligibility test for TTS systems is lacking, although some effort has begun to address this problem [19.42]. For overall quality evaluation, the International Telecommunication Union (ITU) recommends a specific method [19.43] that seems appropriate for testing naturalness. A five-point (or more) rating scale is employed for testing characteristics such as *overall impression*, *listening effort*, *comprehension*, etc. Another option is to ask listeners to state their preference of one option/system over another (A/B tests). Finally, selecting appropriate test material is also relevant. For example, unit-selection systems may sound very good for short sentences spoken in isolation, but show weaknesses when longer paragraphs of text are being rendered. When having a bake-off of different TTS systems, selecting test material appropriate for the intended application seems optimal. For more details on quality testing of TTS systems, see, for example [19.44].

19.5 Conclusions

This chapter summarized the basic principles used in speech synthesis as an introduction to more-detailed chapters later in this book. It is obvious that text-to-speech systems have come a long way towards delivering high-quality output to listeners.

We divided speech synthesis systems into two distinct parts, the text-processing frontend and the speech signal-processing backend. We identified document structure detection, text normalization, interpretation of text markup, and linguistic analysis as frontend tasks. Creating synthetic speech from a symbolic representation is the task of the backend. Here, we touched on articulatory synthesis, formant synthesis, and concatenative synthesis, the latter including unit-selection synthesis. In terms of speech representations and related speech signal-processing algorithms, we highlighted TD-PSOLA, LPC-based synthesis, and sinusoidal synthesis. We then explored transformation techniques for modifying the speech signal either in prosody, or in speaker identity or different speaking styles/emotions. Finally, we touched on evaluation issues for accuracy, intelligibility, and naturalness.

It should be noted that we are still far from delivering perfect synthesis for all possible applications. However, one can note that some of today's systems are now so good that listeners may be fooled into believing that they are listening to recordings of human speakers, in particular, when the system is fine-tuned for a specific application with a limited task domain. For best results, the frontend and the backend of a text-to-speech system can be optimized for a given application. For example, including and maintaining a pronunciation dictionary of names of all family names of telephone subscribers could be essential for using speech synthesis in an automated directory assistance application. Capturing desired voice characteristics (*persona*) from a voice talent that is being recorded for a unit-selection synthesis voice database could be essential for customers accepting an automated dialog system that speaks with a synthetic voice. Much current and future research will focus on improving speech synthesis further with a special focus on automating and streamlining most of the tedious manual processes involved in creating high-quality systems.

References

19.1 J.L. Flanagan: *Speech Analysis, Synthesis and Perception* (Springer, Berlin, Heidelberg 1972) pp. 204–210, http://www.haskins.yale.edu/featured/heads/SIMULACRA/kempelen.html

19.2 H. Dudley, R.R. Riesz, S.A. Watkins: A synthetic speaker, J. Franklin Inst. **227**, 739–764 (1939), http://www.bell-labs.com/org/1133/Heritage/Vocoder/

19.3 W3C Standard Generalized Markup Language: http://www.w3.org/MarkUp/SGML/

19.4 W3C Speech Synthesis Markup Language Version 1.0: http://www.w3.org/TR/2003/CR-speech-synthesis-20031218/ (http://www.xml.com/pub/a/2004/10/20/ssml.html)

19.5 J.P. van Santen: Timing. In: *Multilingual Text-to-Speech Synthesis – The Bell Labs Approach*, ed. by R. Sproat (Springer, New York 1998) pp. 115–139

19.6 D.H. Klatt: Linguistic use of segmental duration in English: Acoustic and perceptual evidence, J. Acoust. Soc. Am. **59**, 1208–1221 (1976)

19.7 A. Schweitzer, B. Moebius: Exemplar-based production of prosody: Evidence from segment and syllable durations, Proc. Speech Prosody 2004 (Nara), ed. by B. Bel, I. Marlien (ISCA, Grenoble 2004)

19.8 K. Silverman, M. Beckman, J. Pitrelli, M. Ostendorf, C. Wightman, P. Price, J. Pierrehumbert, J. Hirschberg: TOBI: A standard for labeling English prosody, Proc. ICSLP'92 Banff (1992) pp. 867–870

19.9 H. Fujisaki: The role of quantitative modeling in the study of intonation, Proc. Int. Symp. Japanese Prosody (1992) pp. 163–174

19.10 G. Richard, M. Liu, D. Sinder, H. Duncan, Q. Lin, J. Flanagan, S. Levinson, D. Davis, S. Slimon: Numerical simulations of fluid flow in the vocal tract, Proc. of Eurospeech Madrid (1995) pp. 18–21

19.11 M. Stone: A three-dimensional model of tongue movement based on ultrasound and x-ray microbeam data, J. Acoust. Soc. Am. **87**, 2207–2217 (1990)

19.12 T. Baer, J.C. Gore, L.C. Gracco, P.W. Nye: Analysis of vocal tract shape and dimensions using magnetic resonance imaging: Vowels, J. Acoust. Soc. Am. **90**, 799–828 (1991)

19.13 C.H. Coker: A model of articulatory dynamics and control, Proc. IEEE **64**, 452–459 (1976)

19.14 E.L. Saltzman, K.G. Munhall: A dynamical approach to gestural patterning in speech production, Ecol. Psychol. **1**(4), 333–382 (1989)

19.15 J. Schroeter, M.M. Sondhi: Speech coding based on physiological models of speech production. In: *Advances in Speech Signal Processing*, ed. by S. Furui, M.M. Sondhi (Marcel Dekker, New York 1991) pp. 231–268

19.16 J.D. Markel, A.H. Gray: *Linear Prediction of Speech* (Springer, New York 1976)

19.17 D.H. Klatt: Software for a cascade/parallel formant synthesizer, J. Acoust. Soc. Am. **67**, 971–995 (1980)

19.18 H.M. Hanson, K.N. Stevens: A quasiarticulatory approach to controlling acoustic source parameters in a Klatt-type formant synthesizer using HLsyn, J. Acoust. Soc. Am. **112**, 1158–1182 (2002)

19.19 J.P.H. van Santen: Combinatorial issues in text-to-speech synthesis, EuroSpeech '97 5th European Conference on Speech Communication and Technology **5**, 2511–2514 (1997)

19.20 J.P. Olive: Rule synthesis of speech from diadic units, Proc. ICASSP **77**, 568–570 (1977)

19.21 O. Fujimura, J. Lovins: Syllables as concatenative phonetic elements. In: *Syllables and Segments*, ed. by A. Bell, J.B. Hooper (North-Holland, New York 1978) pp. 107–120

19.22 J. Olive, J. van Santen, B. Möbius, C. Shih: Synthesis. In: *Multilingual Text-to-Speech Synthesis – The Bell Labs Approach*, ed. by R. Sproat (Kluwer Academic, Dordrecht 1998), Chap. 7

19.23 Y. Sagisaka: Speech synthesis by rule using an optimal selection of non-uniform synthesis units, Proc. ICASSP **88**, 679–682 (1988)

19.24 A. Hunt, A.W. Black: Unit selection in a concatenative speech synthesis system using a large speech database, Proc. ICASSP **96**, 373–376 (1996)

19.25 A.W. Black, N. Campbell: Optimising selection of units from speech databases for concatenative synthesis, ESCA Eurospeech **95**, 581–584 (1995)

19.26 L. Rabiner, B.H. Juang: *Fundamentals of Speech Recognition* (Prentice-Hall, Englewood Cliffs 1993) pp. 339–341

19.27 J. Vepa, S. King: Join cost for unit selection speech synthesis. In: *Text-to-Speech Synthesis – New Paradigms and Advances*, Professional Technical Reference, ed. by S. Narayanan, A. Alwan (Prentice-Hall, Upper Saddle River 2004) pp. 35–62, Chap. 3

19.28 E. Eide, R. Bakis, W. Hamza, J.F. Petrelli: Toward expressive synthetic speech. In: *Text-to-Speech Synthesis – New Paradigms and Advances*, Professional Technical Reference, ed. by S. Narayanan, A. Alwan (Prentice-Hall, Upper Saddle River 2004) pp. 219–248, Chap. 11

19.29 A.W. Black, P. Taylor: Automatically clustering similar units for unit selection in speech synthesis, Proc. Eurospeech **97**, 601–604 (1997)

19.30 K. Tokuda, H. Zen, A.W. Black: An HMM-based approach to multilingual speech synthesis. In: *Text-to-Speech Synthesis – New Paradigms and Advances*, Professional Technical Reference, ed. by S. Narayanan, A. Alwan (Prentice-Hall, Upper Saddle River 2004) pp. 135–153, Chap. 7

19.31 T. Dutoit: *An Introduction to Text-to-Speech Synthesis* (Kluwer Academic, Dordrecht 1997)

19.32 E. Moulines, F. Charpentier: Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones, Speech Commun. **9**(5–6), 453–467 (1990)

19.33 M. Macchi, M.J. Altom, D. Kahn, S. Singhal, M. Spiegel: Intelligibility as a function of speech coding method for template-based speech synthesis, Proc. Eurospeech **93**, 893–896 (1993)

19.34 W. Kleijn, K. Paliwal (Eds.): *Speech Coding and Synthesis* (Elsevier, Amsterdam 1995)

19.35 T.F. Quartieri, R.J. McAulay: Shape invariant time-scale and pitch modification of speech, IEEE Trans. Signal Process. **40**(3), 497–510 (1992)

19.36 Y. Stylianou: Applying the harmonic plus noise model in concatenative speech synthesis, IEEE Trans. Speech Audio Process. **9**(1), 21–29 (2001)

19.37 H. Kawahara, I. Masuda-Katsuse, A. de Cheveigne: Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds, Speech Commun. **27**(3-4), 187–207 (1999)

19.38 M. Abe, S. Nakamura, K. Shikano, H. Kuwahara: Voice conversion through vector quantization, Proc. IEEE ICASSP **88**, 655–658 (1990), S14.1

19.39 I. Stylianou: *Modèles Harmoniques plus Bruit combines avec des Méthodes Statistiques, pour la Modication de la Parole et du Locuteur*, Doctoral Thesis (Ecole Nationole Supérieure des Télécommunications, Paris 1996), in French

19.40 A. Kain, M. Macon: Spectral voice conversion for text-to-speech synthesis, Proc. IEEE ICASPP **98**, 285–288 (1998)

19.41 M.F. Spiegel, M.J. Altom, M.J. Macchi: Comprehensive assessment of the telephone intelligibility of synthesized and natural speech, Speech Commun. **9**, 279–291 (1990)

19.42 A. Syrdal: Development of a standard for the evaluation of intelligibility of text-to-speech synthesis systems by ANSI Accredited Standards Committee S3, Bioacoustics, working group S3/WG 91, Text-to-Speech Synthesis Systems, Personal communication (2007)

19.43 ITU-T: *A Method for Subjective Performance Assessment of the Quality of Speech Output Devices* (International Telecommunications Union, Geneva 1994), Recommendation P.85

19.44 Y.V. Alvarez, M. Huckvale: The reliability of the ITU-T P.85 standard for the evaluation of text-to-speech systems, Proc. ICSLP **2002**, 329–332 (2002)

21. Corpus-Based Speech Synthesis

T. Dutoit

In this chapter, we present the main trends in corpus-based speech synthesis, assuming a stream of phonemes and prosodic target as input. From the early diphone-based speech synthesizers to the state-of-the art unit-selection-based synthesizers, to the promising statistical parametric techniques, we emphasize the engineering trade-offs that arise when designing such systems.

In particular, we examine the mathematical foundations of available methods for modifying the fundamental frequency and the duration of speech units for concatenative synthesis, as well as for smoothing discontinuities at concatenation points. For each of these problems, we analyze time- and frequency-domain processing, using algorithms such as time-domain pitch-synchronous overlap-add (TD-PSOLA), multiband resynthesis overlap-add (MBROLA), and the harmonic-plus-noise model (HNM).

We then provide a comprehensive description of how and why concatenative speech synthesis has progressively adopted large speech corpora, using the principle of context-oriented clustering as a smooth transition from fixed inventory synthesis to unit selection and statistical parametric synthesis.

Our description of unit selection emphasizes important issues related to the definition of optimal target and concatenation costs, as well as to the design of the speech corpus (including memory cost issues) and the reduction of computational costs.

We conclude the chapter with the mathematical framework underlying HMM-based speech synthesis and an outline of its main perspectives.

21.1 Basics

Text-to-speech (TTS) synthesis is often seen by engineers as an easy task compared to speech recognition. In an international conference on speech processing, a famous scientist once brought up a tube of toothpaste (whose brand was in fact Signal) and, pressing it in front of the audience, he coined the words: *This is speech synthesis; speech recognition is the art of pushing the toothpaste back into the tube* ... It is true, indeed, that it is easier to create a bad, first trial TTS system than to design a rudimentary speech recognizer. After all, recording numbers up to 60 and a few words (*it is now*, *am*, *pm*) and being able to play them back in a given order provides the basis of a working talking clock, while trying to recognize such simple words as *yes* or *no* immediately implies some more-elaborate signal processing. If speech synthesis was really *that* simple, however, one could only blame the text-to-speech (TTS) research and development (R&D) community for not having been able to massively produce a series of talking consumer products as early as the 1980s.

The major point is that users are generally much more tolerant to automatic speech recognition (ASR) errors than they are willing to listen to unnatural speech. There is magic in a speech recognizer that transcribes continuous radio speech into text with a word accuracy as low as 50%; in contrast, even a perfectly intelligible speech synthesizer is only moderately tolerated by users if it delivers nothing other than *robot voices*.

This importance of *naturalness* versus *intelligibility* is actually very typical of the synthesis of natural signals (as opposed to their recognition). One could thus advantageously compare speech synthesis to the synthesis of human faces: while it is quite easy to sketch a cartoon-like drawing that will be unanimously recognized as a human face, it is much harder to paint a face that will be mistaken for a photograph of a real human being. To a large extent you can modify the size, position, and orientation of most of the elements of a hand drawing without breaking the intelligibility barrier (just think of the cubists, etc.), but even a slight change to a photorealistic painting will immediately make the complete work look like what it actually is: a painting of a face, not a real photograph of it.

Delivering intelligibility *and* naturalness has thus been the holy grail of speech synthesis research for the past 30 years. Speech expressivity is now increasingly seen as an additional objective to reach (Chap. 25). Add to it that engineering costs (computational cost, memory cost, design cost for having another synthetic voice or another language) have always had to be taken into account, and you will start to have an approximate picture of the challenges underlying text-to-speech synthesis.

This chapter outlines the specific problems encountered when trying to reach these goals, and shows how today's most-advanced solutions benefit from *corpus-based* speech synthesis techniques. We basically distinguish three approaches: *concatenative synthesis based on a fixed inventory* (Sect. 21.2), in which speech is obtained by gluing speech chunks taken from a limited-sized, carefully prepared, unit inventory; *concatenative synthesis based on unit selection* (Sect. 21.3), in which the units inventory is several orders of magnitude larger; and *statistical parametric synthesis* (Sect. 21.4), in which speech synthesis implies the adequate concatenation of statistical models of speech units.

21.2 Concatenative Synthesis with a Fixed Inventory

Producing speech samples automatically does not merely reduce to the playback of a sequence of prerecorded words or phonemes, due to *coarticulation*. Coarticulation results from the fact that each articulator moves continuously from the realization of one phoneme to the next. It appears even in the most carefully uttered speech. In fact, it *is* speech. Thus, producing intelligible speech requires the ability to produce continuous, coarticulated speech.

More generally, transients in speech are more important for intelligibility than stable segments [21.1, 2], while modifying stable segments (e.g., the center of vowels) can very easily affect naturalness. Speech, indeed, is never really periodic, nor stable. Even sustained vowels exhibit small frequency and amplitude variations (respectively termed *jitter* and *shimmer*), and have substantial inharmonic components due to non-complete closure of the vocal folds after the so-called *glottal closure instant*; the presence of these noise-like components is correlated with specific events within the pitch period. As a result, the intuitive concept of *adding a bit of noise* to intonation curves, to amplitude curves, or to voiced speech waveforms in order to *make them sound more natural* merely leads to more-noisy synthetic speech: inharmonicity in voiced speech is not pure randomness.

Concatenative synthesis techniques try to deliver the expected intelligibility and naturalness of synthetic speech by gluing together speech chunks that embody natural coarticulation, jitter, shimmer, and inharmonicity.

21.2.1 Diphone-Based Synthesis

Constraining speech chunks to embody coarticulatory effects can be achieved to some extent by using *diphones* (or *dyads*) as basic units. A diphone is a speech segment that starts in the middle of the stable part (if any) of a phoneme and ends in the middle of the stable part of the next phoneme [21.3]. Diphones therefore have the same average duration as phonemes (about 100 ms), but if a language has N phonemes, it typically has about N^2 diphones, or slightly fewer in practice since not all diphones are encountered in natural languages. This leads to a typical diphone database size of 1500 diphones

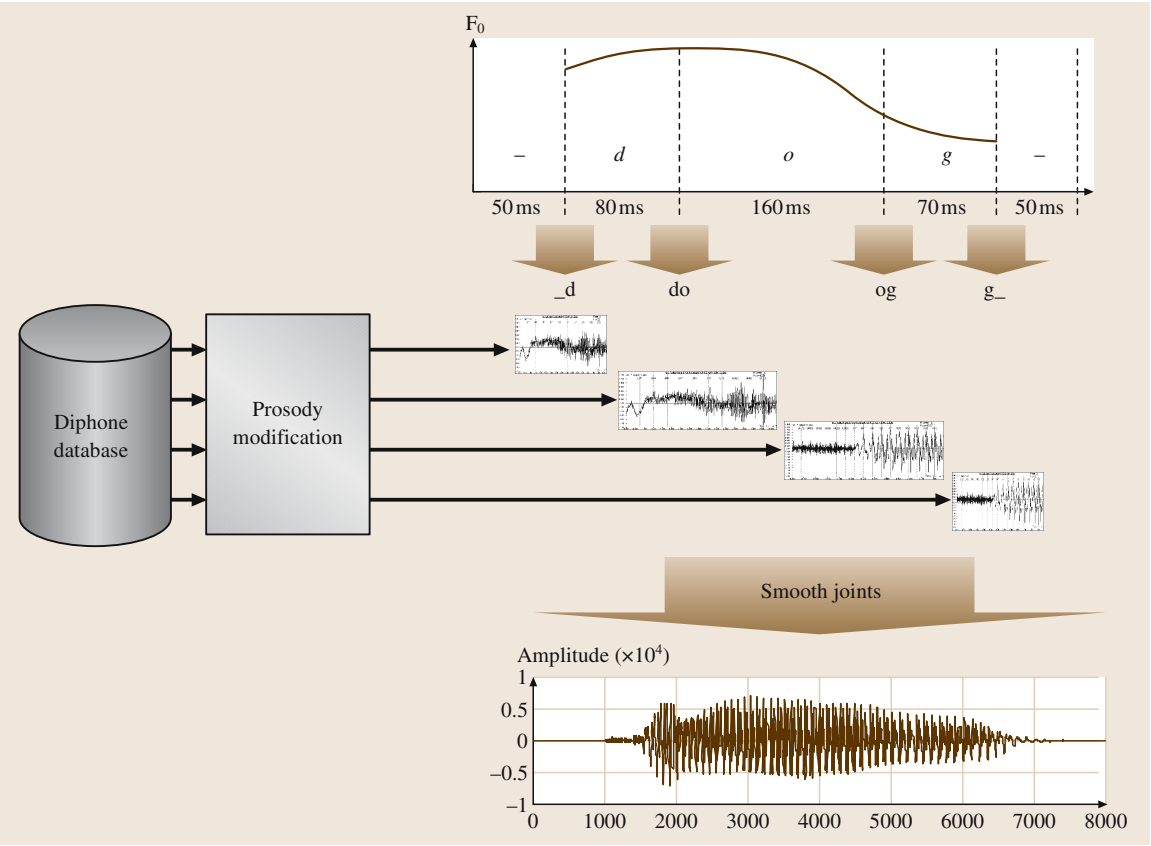


Fig. 21.1 A schematic view of a diphone-based speech synthesizer. In order to produce the word 'dog' with given phoneme durations and fundamental frequency, the diphone inventory (or database) is queried for the corresponding sequence of diphones. The duration and pitch of each diphone is modified, and some smoothing is applied on joints

(about 3 min of speech, i. e., about 5 MB for speech sampled at 16 kHz/16 bits).

In order to create such a synthesizer, one needs to set up a list of the required diphones; a corresponding list of words is carefully completed in such a way that each segment appears at least once (twice is better, for security). Unfavorable positions, such as those inside strongly stressed syllables or in strongly reduced (over-coarticulated) contexts, are usually excluded. A corpus is then read by a professional speaker (avoiding speaking-style variations as much as possible, and even possibly without large pitch variations, so as to facilitate speech analysis) and digitally recorded and stored. The elected segments are spotted, either manually with the help of signal visualization tools, or automatically thanks to segmentation algorithms, the decisions of which are checked and corrected interactively. Segment waveforms are then collected into a diphone inventory (or database). This operation is performed only once.

At run time, once the synthesizer (Fig. 21.1) receives some phonetic input (phonemes, phoneme duration, pitch) from the natural language processing (NLP) module, it sets up a list of required diphones, together with their required duration and fundamental frequency contour. The available diphones would only match these prosodic requests by chance, since they have generally been extracted from words and sentences which may be completely different from the target synthetic sentence. Some *prosody modification* is therefore necessary. How this is achieved is examined in more detail in Sect. 21.2.2. Additionally, since the diphones to be concatenated have generally been extracted from different words – that is, in different phonetic contexts – the end of one diphone and the beginning of the next often do not fully match in terms of amplitude and/or spectral

envelope. Part of this problem can be solved by *smoothing* individual pairs of successive segments. Smoothing will be discussed in Sect. 21.2.3. In Sect. 21.2.4, we will see what solutions are available for expanding the unit inventory.

21.2.2 Modifying Prosody

Prosody refers to the properties of a speech signal that are related to audible changes in pitch, syllable length, and loudness (Chap. 23).

Of the three parameters, pitch is clearly the most sensitive. Even slightly modifying the shape of the pitch curve readily breaks the naturalness of a speech segment. This effect is still more audible when pitch modification ratios on both sides of concatenated units differ too much. Applying a constant pitch-modification ratio on a complete segment (as in [21.5]) does less harm, provided that the ratio is not too high (typically ≤ 2) or too low (typically ≥ 0.5). Applying duration modification to diphones to match the required phoneme length is less sensitive, provided the ratio is reasonable (≤ 2 and ≥ 0.5). Modifying the loudness of diphones on the fly is trivial, but generally not much used in diphone-based synthesis. An offline equalization step is generally preferred, in which related endings of diphones are given similar amplitude spectra, the difference being distributed over their neighborhood

Neither pitch nor duration modification is a straightforward operation. Simply resampling speech data (creating additional speech samples by interpolation of original ones while keeping the same sampling frequency for playing them) does modify duration, but also the spectral envelope. All formants become compressed/dilated along the frequency axis, which has exactly the same audible effect as that of changing the *pitch* wheel of a tape recorder: human voices turn into doggy or mousy voices.

Two main classes of algorithms have been proposed for prosody modification, namely *time-domain* and *frequency-domain* algorithms.

Time-Domain Prosody Modification

The distinctive feature of the time-domain pitch-synchronous overlap-add (TD-PSOLA) algorithm [21.6] is that it makes it possible to perform both the required pitch and duration modifications directly on continuous waveforms, without any parametric speech modeling.

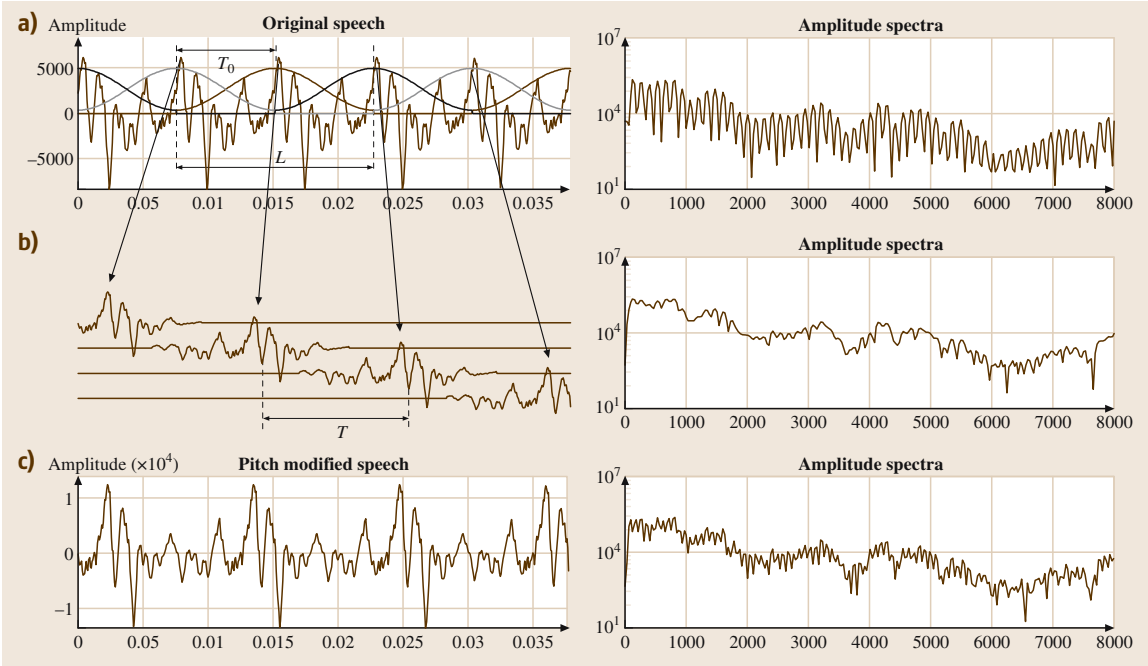


Fig. 21.2a–c The TD-PSOLA reharmonization process. The pitch-modified waveform **(c)** has the same spectral envelope as the original waveform **(a)**. The center plot **(b)** shows the OLA frames and the amplitude of their Fourier transform, which is approximately the spectral envelope of the initial signal (after [21.4])

As a matter of fact, if we restrict ourselves to a purely periodic signal $s(n)$, it is possible to obtain a perfect pitch-modified version of $s(n)$ by summing overlap-add (OLA) frames $s_i(n)$, extracted pitch-synchronously from $s(n)$ with a weighting window $w(n)$, and changing the time shift between frames from the original pitch period T_0 to the desired period T:

$$s_i(n) = s(n)w(n - iT_0)\,, \tag{21.1}$$

$$\tilde{s}(n) = \sum_{i=-\infty}^{\infty} s_i[n - i(T - T_0)]\,. \tag{21.2}$$

If $T \neq T_0$, the operation results, according to the Poisson formula, in a reharmonization of the spectrum of $s_i(n)$ (which, if we assume perfect periodicity, does not depend on i when the original voiced sound is purely periodic) with fundamental frequency $1/T$:

$$\text{if } s_i(n) \overset{F}{\longleftrightarrow} S_i(\omega)$$

$$\text{then } \tilde{s}(n) \overset{F}{\longleftrightarrow} \frac{2\pi}{T} \sum_{i=-\infty}^{\infty} S_i\left(i\frac{2\pi}{T}\right)\delta\left(w - i\frac{2\pi}{T}\right)\,. \tag{21.3}$$

Consequently, provided $w(n)$ can be chosen so that the spectrum of $s_i(n)$ closely matches the spectral envelope of $s(n)$, (21.2) provides a simple and efficient way to change the pitch of a periodic signal (Fig. 21.2).

Obtaining an estimate of the spectral envelope of $s(n)$ through $s_i(n)$ is obtained when the length of the weighting window applied to each OLA frame is approximately twice the local pitch period (Fig. 21.3), which implies that this length is signal dependent. Notice also that $w(n)$ needs to be smoothly zero-ended, so as to avoid clicks at each OLA operation. Typically, Hanning windows are used. Another practical constraint is that F_0 modification produces its best quality when OLA frames are centered on the glottal closure instant, or at least on an instant of important speech excitation (as is the case in Fig. 21.2). As a result, TD-PSOLA requires the prior computation (or the semiautomatic setting) of *pitch marks* on all speech units. However, computing glottis closure instants (the only reference instant available in voiced speech) has two major drawbacks: finding glottal closure instants is not very robust if performed automatically, and it is time consuming if performed manually. A possible alternative to this problem is to measure glottal activity directly, with the help of an electroglottograph (EGG), instead of trying to deduce it from speech. Another solution is to use the local center of gravity of speech as the center of the OLA frames [21.7].

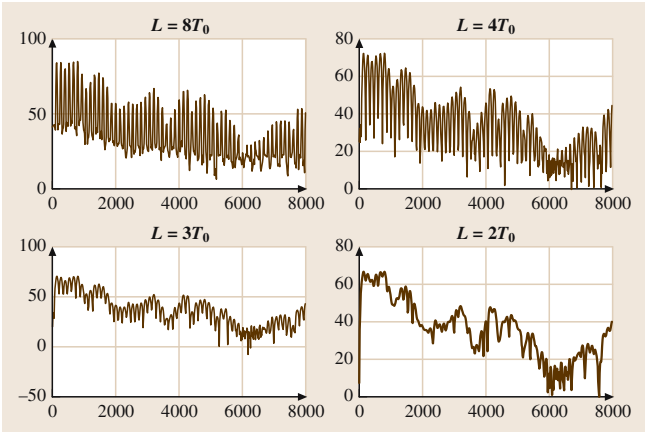


Fig. 21.3 Amplitude spectra of the OLA frames extracted from the vowel [a] for several values of the weighting window length L (F_s: 16 kHz; window: Hamming). Choosing $L = 2T_0$ provides an approximation of the spectral envelope

For simultaneous duration and pitch modification, pitch-synchronous frames must be used. Let us first assume that, prior to any processing, we associate with each analysis pitch mark η^i the value of its local pitch period T_0, so that a pair (η^i, T_0^i) suffices to define the corresponding analysis OLA frame $s_i(n)$. What we want to do is a mapping between pitch marks on the analysis time axis t and corresponding pitch marks on the synthesis time axis t', where the relationship between t and t' is defined by a function $t'(t)$, so as to impose pitch on the synthesis time axis to be equal to some function

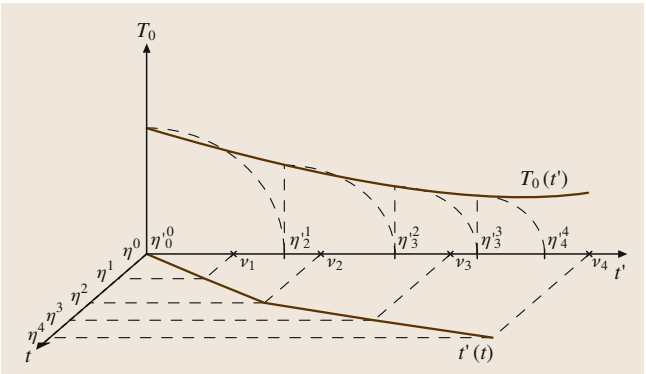


Fig. 21.4 Pitch and timing modifications with TD-PSOLA. Starting from five analysis OLA frames on the analysis time axis (t), four synthesis pitch marks are positioned on the synthesis time axis (t'), and each is respectively associated to an analysis frame. In the example shown, frame 1 has been eliminated, while frame 2 has been duplicated (after [21.4])

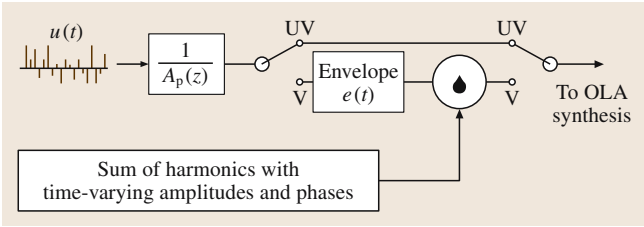


Fig. 21.5 Speech synthesis in the HNM model

$T_0(t')$ (Fig. 21.4). It is easy to derive the position of synthesis pitch marks η'^j: the shift between two successive synthesis pitch marks η'^j and η'^{j+1} is adjusted to the value of the synthesis pitch period at time η'^j: $T_0(\eta'^j)$. By applying the time alignment function $t'(t)$ to each initial pitch mark position η^i, we also obtain the corresponding *virtual pitch marks*, denoted by $\eta^i = t'(\eta^i)$. It is then easy to associate an analysis pitch mark η^i with each synthetic pitch mark η'^j by looking for the closest virtual pitch mark on the synthesis time axis (Fig. 21.4). Given the time distortion introduced by $t'(t)$, some analysis pitch marks are not associated with synthesis pitch marks, while others are duplicated. A list of synthesis parameters is finally created, defined by triplets (η'^j, η^i, T_0^i) in which η'^j gives the position of each synthesis OLA frame and η^i, T_0^i refers to the analysis OLA frame associated to the current synthesis frame. These triplets are indicated as ${\eta'^j}_i$ in Fig. 21.4.

Notice that duration-only modification can be achieved in the time domain by simply replicating or eliminating some OLA frames. In this case, OLA frames need not be pitch-synchronous (as in the waveform similarity overlap add (WSOLA) method [21.8], and the pointer interval controlled overlap and add (PICOLA) algorithm [21.9], implemented in the ISO MPEG-4 audio standard). When two noncontiguous frames are overlapped, a cross-correlation analysis is performed to find the optimal lag or lead between the frame centers of the source waveforms. In order to prevent the algorithm from introducing artificial short-term autocorrelation in the synthesis signal when unvoiced OLA frames are *n*-plicated (i. e., when speech is very much slowed down), every other duplication of an unvoiced frame can be reversed in time (which does not change its spectral content). With such an artifice, speech can reasonably be slowed down by a factor of four, even though some tonal effect is encountered in voiced fricatives, which typically combine voiced and unvoiced frequency regions and therefore cannot be inaudibly time reversed.

TD-PSOLA has been widely used in diphone-based synthesis, because of its high synthesis quality and extraordinarily low computational load (typically seven operations per sample). Synthesized speech, however, is not perfect: spectral mismatches cannot easily be eliminated at segmental boundaries and tonal quality appears when large prosodic modifications are applied on the concatenated units.

Frequency-Domain Prosody Modification

Sinusoidal approaches and hybrid harmonic/stochastic representations flourished in speech processing as early as the early 1980s [21.10, 11], and have been proposed for speech synthesis [21.7, 12]. In particular, the

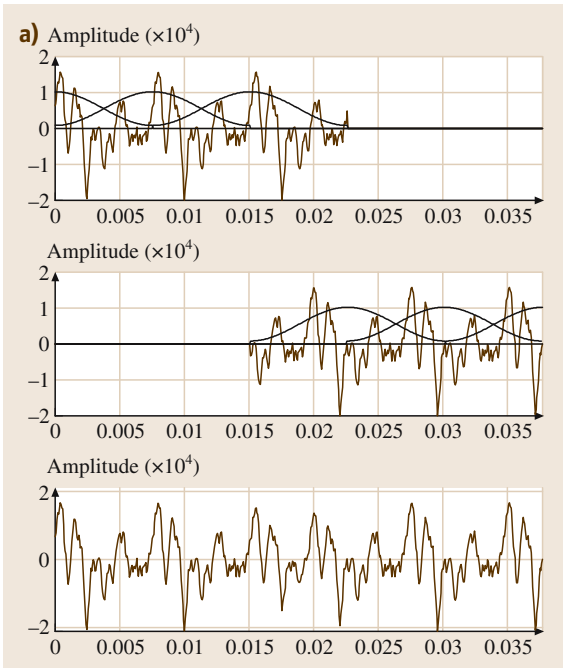


Fig. 21.6a–c *Top*: last frames of the first segment to be concatenated; *center*: first frames of the second one; *bottom*: after OLA; **(a)** Phase mismatch: waveforms are identical but the OLA windows are not centered on the same relative positions within the period. **(b)** Pitch mismatch: both segments have exactly the same spectral envelope, but were pronounced with different pitches. OLA windows are positioned coherently. The synthetic signal is obtained after changing the interval between the frames of the right segment in order to impose a constant pitch. **(c)** Spectral envelope mismatch: the left segment is the diphone 'ma'; the right one is 'am'. The pitch is constant, and the windows are again positioned coherently. The spectral discontinuity is concentrated in one period (after [21.4]) ▶⇒▶▶

harmonic-plus-noise model (HNM) [21.7]) is widely used because of its additional smoothing (Sect. 21.2.3) and speech compression capabilities. In this model (Fig. 21.5), speech is assumed to be composed of two additive components, a harmonic component $h(t)$, and a noise component $n(t)$, corresponding to a separation of the speech spectrum into two bands:

$$\tilde{s}(t) = h(t) + n(t) \,. \tag{21.4}$$

A time-varying maximum voiced frequency determines the limit between the two bands. In the lower band, the signal $h(t)$ is represented as a sum of $K(t)$ harmonically related sine waves with slowly varying amplitudes and frequencies:

$$h(t) = \sum_{k=1}^{K(t)} A_k(t) \cos[k\theta(t) + \phi_k(t)] \tag{21.5}$$

$$\text{with } \theta(t) = \int_{-\infty}^{t} \omega_0(l)\,\mathrm{d}l \,. \tag{21.6}$$

The amplitudes $A_k(t)$ and phases $\phi_k(t)$, as well as the fundamental frequency $\omega_0(t)$ of these sine waves, are estimated from real speech signal by minimizing a weighted time-domain least-squares criterion [21.7]. The estimation is repeated for successive speech frames, extracted from the input speech with constant frame shift S (i. e., non-pitch-synchronously). In other words, the centers of the analysis frames η^i are obtained by

$$\eta^{i+1} = \eta^i + S \,. \tag{21.7}$$

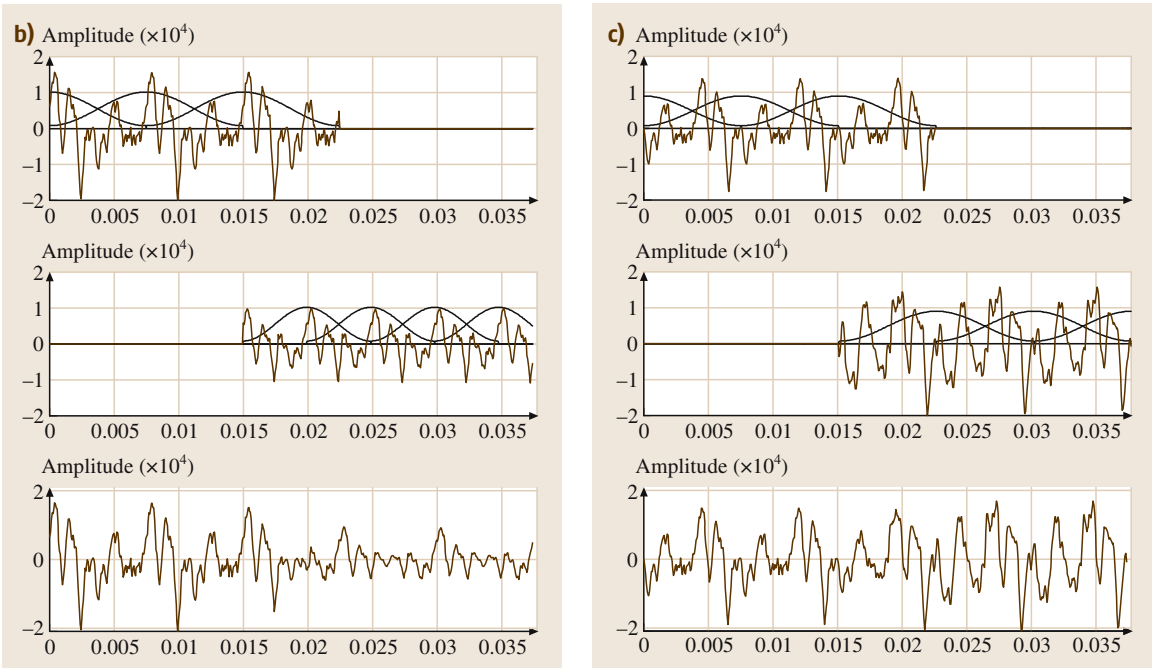
The upper band component, i. e., the noise part $n(t)$, is modeled by filtering a white Gaussian noise $u(t)$ by a time-varying, normalized all-pole filter $h(\tau, t)$ and multiplying the result by a time-domain amplitude envelope function $e(t)$:

$$n(t) = e(t)[h(\tau, t) * u(t)] \,. \tag{21.8}$$

This makes it possible to account for the nonperiodic components of speech, which include frication noise and period-to-period variations of the glottal excitation. In particular, the envelope function is synchronized with the harmonic part. If this is not the case, then the noise part is not perceptually integrated with the harmonic part but is rather perceived as additive, separate noise.

In order to modify the pitch and duration of speech units modeled by HNM, a set of synthesis pitch marks η'^j are first obtained from the η^i through the same operation as in Fig. 21.4. Then, for each synthesis pitch mark found, pitch modification implies estimating amplitudes and phases of harmonics at the new harmonic frequencies by resampling the complex speech spectrum.

The last step is the generation of the synthetic signal itself using the stream of modified



HNM parameters. Synthesis is performed pitch-synchronously using an overlap and add (OLA) process. OLA is performed on synthesis OLA frames $\tilde{s}_i(t)$ computed from (21.3), using the aforementioned pitch-modified amplitudes and phases. Synthesis OLA frames $\tilde{s}(t) = h(t) + n(t) = \sum_i a_i(t)\cos(\phi_i(t)) + \tilde{s}_r(t)$ are made pitch-synchronous by adjusting the phases of their harmonics, so as to impose their center of gravity. In practice, the noise part is also filtered by a high-pass filter with a cut-off frequency equal to the maximum voiced frequency associated with the frame.

21.2.3 Smoothing Joints

Concatenating speech units possibly extracted from different words or phonetic contexts is not straightforward. Done without care, it produces audible clicks, which cannot always be removed by simple OLA between the end of the left unit and the beginning of the right unit. This is due to at least three types of mismatches between successive units, respectively related to phase, pitch, and spectral envelope mismatches [21.13], and illustrated in Fig. 21.6 for the concatenation of vowels [a] sampled at 16 kHz.

Phase Mismatch

Phase mismatch originates in overlap-add frames that are not centered at the same place within the period (Fig. 21.6a); even if the wave shapes to be concatenated are identical, the OLA operation results in an unexpected discontinuity in this case.

With time-domain methods, this can be avoided by adequate pitch-marking of boundary frames (as done in TD-PSOLA for all speech segments). Another solution is to adjust the position of OLA frames on the left and right units so as to maximize the cross-correlation between windowed frames (as in WSOLA), thereby maximizing the chance that OLA frames are synchronous [21.8]. This solution, which corresponds to finding a common relative synchronization point, is easier than absolute synchronization on glottal closure instants, but it is more time-consuming (it is usually performed online, since synchronization points cannot be precomputed and stored for all possible pairs of speech segments). It is also possible to use the aforementioned center of gravity of the OLA frames as an absolute reference point for centering frames [21.7], which can be done offline.

With frequency-domain techniques, phase mismatch can be avoided by making sure that all synthesis OLA frames have the same initial harmonic phases. This is done in the *multiband resynthesis OLA* (MBROLA) technique, a time-domain technique based on speech samples that have preliminarily (once and for all) been edited using frequency-domain modification [21.13]. As a result of this modification, MBROLA frames have identical pitch and harmonic phases. A more-realistic option, used with HNM, is to measure phase mismatches before synthesizing OLA frames, and making sure that the center of gravity of the synthesis OLA frames are synchronized by correcting their phases (this corresponds to imposing the phase of the fundamental on all synthesis OLA frames).

Pitch Mismatch

Pitch mismatch arises when overlapped frames have very different fundamental periods, even if they have identical spectral envelopes and are windowed on similar relative positions within the pitch period. This situation cannot easily be avoided in a database of several thousand segments (Fig. 21.6b).

It is nevertheless possible to minimize this effect by recruiting professional speakers to record the segments database and training them to read the corpus with the most constant pitch possible (TD-PSOLA) or by editing the recorded speech so as to impose constant pitch (MBROLA).

Another solution is to apply frequency-domain pitch modification on the fly before the OLA operation, as in HNM. First, the difference between the pitch value of the last frame on the left and of the first frame on the right of a concatenation point is measured. Then this difference is distributed on several frames on the left and on the right of the concatenation point. This is typically achieved by resampling the complex speech spectrum of these frames and computing new harmonic amplitudes and phases for the new pitch value.

Spectral Envelope Mismatch

Last but not least, *spectral envelope mismatch* is due to coarticulation and speaker variability and appears whenever the speech units to be concatenated have been extracted from rather different phonetic contexts (Fig. 21.6c).

It can be minimized somewhat, by using adjustable unit boundaries [21.14] (i. e., by changing the left and right borders of the diphones on the fly).

The standard implementation of TD-PSOLA, however, cannot smooth out spectral envelope mismatches. As a result, synthesizing highly fluent speech with TD-PSOLA often requires a painstaking trial-and-error process in which the most probable diphone sequences

are systematically tested and diphones are rejected when an important spectral envelope mismatch is encountered. Other versions of the rejected diphones are then extracted from words with a different phonetic context than the initial one; tests are run again, and so on.

A partial solution consists of resynthesizing the speech segment database so as to provide segments with interesting properties which can be taken into account at synthesis time, as in MBROLA. Since all MBROLA OLA voiced frames have the same F_0 and the same initial harmonic phases, it is possible to perform period-based smoothing in the time domain. This feature somehow compensates for the loss of segmental quality due to the resynthesis operation.

It is also possible to achieve spectral envelope smoothing in the frequency domain. Harmonic/stochastic models such as HNM proceed by linearly interpolating harmonic amplitudes (perceptually, discontinuities in the parameters of the noise part are not important and are thus ignored). First, the differences between the amplitudes of each harmonic (between frames on the left and frames on the right of a concatenation point) are measured. Then, these differences are weighted and propagated left and right from the concatenation point. The number of frames used in the interpolation process depends on the variance of the number of harmonics and the size, in frames, of the concatenated speech units.

Such a simple linear interpolation of the spectral envelopes makes formant discontinuities less perceptible. However, if formant frequencies are very different at the left and right of the concatenation point, the problem is not completely solved. A partial solution to this was proposed in [21.15], where smoothing between two units (the unit on the left l and the unit on the right r) is performed with reference to a template speech segment lr (i. e., a natural example of lr taken from continuous speech). The template segment, termed the *fusion unit*, is merged with the end of l and the beginning of r, in the frequency domain.

21.2.4 Up from Diphones

The acoustic units chosen as building bricks for a concatenative synthesizer should obviously exhibit some basic properties: they should account for as many coarticulatory effects as possible; they should be available with all sorts of durations, pitch contours, and voice qualities (so as to alleviate the task of the prosody modification block); they should be easily connectable (without requiring complex rule-based smoothing techniques of formant synthesizers for instance; Chap. 20). Their number should also be as small as possible (to avoid having to prepare, record, segment, and store a large unit inventory), although on the other hand longer units (hence, larger unit inventories) decrease the density of concatenation points, therefore providing better speech quality.

Diphones are a good compromise when the number of units has to be kept really small (i. e., typically for cheap embedded applications). They imply, however, a high density of concatenation points (one per phoneme), which reinforces the importance of an efficient concatenation algorithm. Besides, they can only partially account for the many coarticulatory effects of a spoken language, since these often affect a whole phoneme rather than just its right or left halves independently: formant targets themselves tend to be changed, or a formant starts rising (or falling) during the realization of one phoneme and continues that movement smoothly throughout the ensuing one without any steady state. Such effects are especially obvious when somewhat transient phones, such as liquids and (worst of all) semivowels, are concatenated.

Assimilations (i. e., very strong forms of coarticulation, in which a phonetic trait of a phoneme is completely modifies as a result of the preceding or following phonemes) are still a bigger source of concern. Partial assimilation cases are approximately covered by diphones, while total cases would require the introduction of several realizations per phoneme in the database (as many as there are allophones) and a corresponding number of diphones. Such a strategy is sometimes called *allo-diphone-based synthesis*.

Triphones (which embody a complete phoneme) constitute a variant to the use of allo-diphonic units as a complement to diphones. They can embody the strong dynamic effects quoted above, providing these only affect one phoneme. To a larger extent, associating diphones with tri- and tetraphones to cope with a limited number of very specific contextual effects is often called a *polyphone* approach [21.16]. France Telecom [21.17], for instance, has reported on an enhanced French polyphone database composed of 1290 diphones, 751 triphones, and 288 quadriphones.

Another way of keeping the database size small while taking syllables into account to a degree is to split them into two at their vocalic nucleus, which acts as a coarticulation barrier in most cases. Such demi-syllables (of the form C–V or V–C) have been fully adopted for the synthesis of German, in which long consonant clusters are manifold. Such a combination of

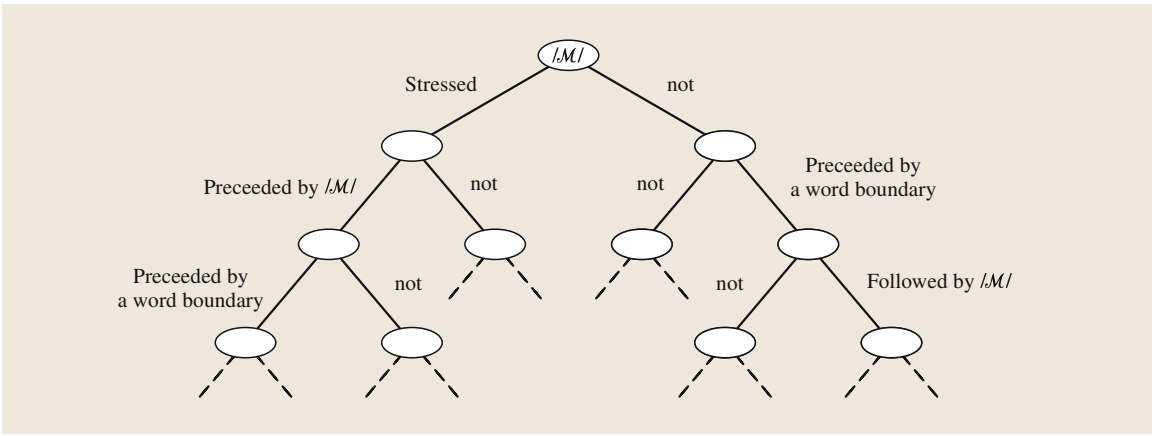


Fig. 21.7 An example of the COC cluster splitting process for occurrences of the phoneme /ɱ/ (after [21.19]). As a result of this process, allophones of /ɱ/ are detected, and explained

HAlf syllables, DIphones, and sufFIXes is the basis of the HADIFIX system [21.18].

Notice, finally, that all the types of speech units proposed above are based on some pre-established phonetic knowledge of coarticulatory or assimilation effects and where they mostly occur. In contrast, nothing truly impedes them from being directly automatically derived on the basis of some automatic analysis of the distribution of their spectral envelopes. This has led to the proposal of a corpus-based, automatic unit set design algorithm termed *context-oriented clustering* (COC) [21.19].

COC builds a decision tree for each phoneme, which automatically *explains* the variability in the acoustic realization of this phoneme in terms of contextual factors. Starting from a speech database with phonetic labels, it constitutes initial clusters composed of all the speech units (allophones in this case) with the same phonetic label (Fig. 21.7). The context tree is then automatically derived by a greedy algorithm. At each step, the cluster with the highest variance (and with more than a minimum number of segments) is split into two classes on the basis of a partition of a *context factor*, and provided the resulting split leads to an average subcluster variance that is significantly lower than the initial one. Intracluster variances are computed on segments after a normalization of their duration, based on a linear warping technique. Notice that COC is very similar to the tree-based decomposition of classification and regression tree (CART) techniques. The major difference is that CARTs are trained in a *supervised* way (they account for pre-established decisions), while COC methods are *unsupervised*. Hence the use of an acoustic distance to obtain the best contextual factor, in place of the maximum mutual information principle of CARTs. Possible context factors used (i. e., the factors that are submitted to the tree-building algorithm to partition clusters) are the phonetic labels of neighboring phones up to a given distance from the central one, as well as stress and syntactic boundaries. Broad phonetic classes can also be used as contextual factors, so as to reduce the size of the decision tree and to cope with the problem of data sparsity.

It is important to understand that the output of this COC method, when used for concatenative synthesis using fixed inventories, is basically the resulting tree. Once the tree has been built, the content of its leaves (the speech units taken from the speech database and associated to leaves) is reduced to a single representative speech unit (even possibly rerecorded on purpose). This technique can thus be seen as an automatic method for building a list of allophones for each phoneme (i. e., variants of a given phoneme due to a specific phonetic/syntactic/stress context).

This idea was further extended to subphonetic units by first segmenting some large speech corpus using hidden Markov model (HMM)-based forced alignment, and then applying a similar context clustering method to the contents of the HMM states [21.20].

While speech synthesis based on the concatenation of units taken from a fixed inventory is clearly intelligible (and has been so from its very beginning [21.21]), it cannot really be termed natural, whichever set of units are chosen. In the next section, we examine how naturalness can be approached, although sometimes at the expense of intelligibility.

21.3 Unit-Selection-Based Synthesis

The lack of naturalness of speech synthesized from a fixed inventory (i. e., with the constraint of having only one instance of each possible concatenative unit stored in the unit database) has two main causes. First, this strategy unavoidably biases the unit recording step towards the choice of a somewhat overarticulated instance, which will fit in most contexts and lead to the best intelligibility score. Additionally, as we have seen in the previous section, signal-processing tricks are needed to adapt the pitch and duration of each unit (if only to be able to synthesize stressed and unstressed versions of the same syllable). Some signal degradation is the price to pay for this processing.

It will thus come as no surprise that speech synthesis made a huge step forward when researchers started keeping several instances of each unit, and provided means of *selecting*, at run time, which instance to use for synthesizing a given sentence.

21.3.1 Selecting Units

After first trials in this direction in the *ν*-Talk TTS system [21.22], with mixed results, the foundations of unit-selection-based speech synthesis was laid in [21.23, 24], which quickly led to ATR's CHATR TTS system, and to AT&T's NextGen TTS system [21.25]. At

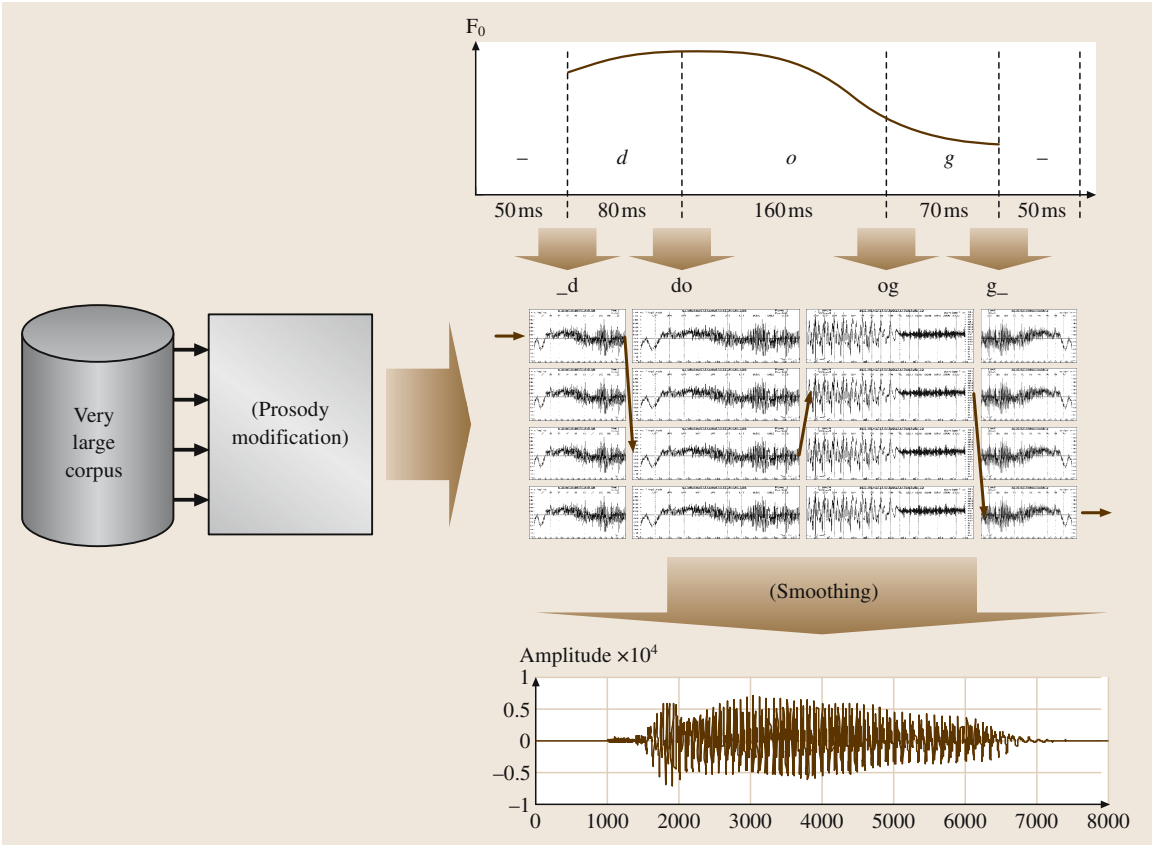


Fig. 21.8 A schematic view of a unit-selection-based speech synthesizer. The prosody modification and smoothing modules have been included between parentheses, since they are not always implemented. As a matter of fact, since this approach uses very large speech corpora, it is often possible to find speech units that naturally join smoothly while exhibiting prosodic features close to what is expected. Notice that, unlike suggested in Fig. 21.8, unit-selection-based synthesis systems do not systematically use diphone units. Diphones have been shown here for the sake of easy comparison with Fig. 21.1

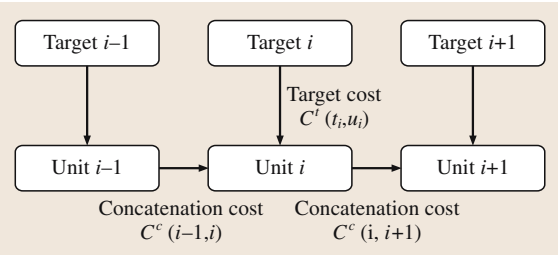


Fig. 21.9 Target and concatenation costs

run time, given a phoneme stream and target prosody for an utterance, the unit selection algorithm selects from a speech corpus an optimum set of acoustic units, i. e., the one that *best* matches the target specifications (Fig. 21.8, to be compared to Fig. 21.1). Units may still be diphones, although phonemes and subphonetic units such as half-phonemes can be used as alternatives when complete diphones are not available with the required pitch and duration. For every *target unit* t_i required (for example, for every diphone to be synthesized), the selection algorithm first proposes a list of *candidate units* from the speech database, each in a different context (and in general not exactly in the same context as the target unit). The final choice among candidates is based on minimizing the sum of two cost functions: a *target cost* $C^t(t_i, u_i)$, which is an estimate of the difference between a candidate unit u_i and the target unit t_i it is supposed to represent, and a *concatenation cost* $C^c(u_{i-1}, u_i)$, which is an estimate of the quality of the joint between candidate units u_{i-1} and u_i for two consecutive targets (Fig. 21.9). The best sequence of n units u_1^n for a given sequence of n targets t_1^n is the one that minimizes the total cost:

$$C(t_1^n, u_1^n) = \sum_{i=1}^{n} C^t(t_i, u_i) + \sum_{i=2}^{n} C^c(u_{i-1}, u_i) + C^c(S, u_1) + C^c(u_n, S) , \quad (21.9)$$

where S denotes the target for a silence. This best sequence is found using a Viterbi search.

When good candidate units cannot be found with the correct pitch and/or duration, prosody modification can be applied. Also, as candidate units usually do not concatenate smoothly (unless a sequence of such candidate units can be found in the speech corpus, perfectly matching the target requirement), some smoothing can be applied (with the methods exposed in Subsects. 21.2.2 and 21.2.3). The latest synthesizers, however, tend to avoid prosodic modifications and smoothing, assuming that the speech unit inventory is rich enough for the requested units to be available with approximately correct pitch and duration [21.26]. This assumption is sometimes referred to as *take the best to modify the least*.

The challenges in unit-selection synthesis are related to the development of efficient target and concatenation costs for finding the *best* path in the candidate unit lattice, to the definition of optimal speech corpus, and to the engineering of efficient search algorithms. These issues are addressed in the next subsections.

21.3.2 Target Cost

The optimal target cost should estimate the *perceptual* distance from a specific inventory unit to the desired target unit. Candidate units are characterized by a feature vector. For each target unit in the utterance to be synthesized, appropriate feature values are predicted. The target cost $C^t(t_i, u_i)$ for a given candidate u_i to a given target t_i is calculated as the sum of weighted feature vector differences between the candidate and target. These differences are p target subcosts $C_j^t(t_i, u_i)(j = 1, \ldots, p)$:

$$C^t(t_i, u_i) = \sum_{j=1}^{p} w_j^t C_j^t(t_i, u_i) . \quad (21.10)$$

Target feature weights w_j^t are trained during the creation of a TTS voice, to optimize the mapping from the feature vector differences between units to the perceptual differences between these units.

Initial experiments on unit selection (such as [21.23]) proposed to use a combination of symbolic features such as phonetic context or stress, and numerical features such as duration and F_0. The list of features was later biased towards symbolic (or *phonological*) features, which leave more freedom for the ultimate choice of units while keeping most of the intention of the virtual speaker [21.27]. This is especially useful for intonation, whose numerical prediction is particularly difficult. Instead of trying to predict F_0 values from text, most TTS systems currently predict a sequence of abstract *tonal* units, which are only *qualitatively* related to the intended intonative movements. As a result, intonation is obtained as a by-product of unit selection itself, provided that the correct linguistic features are used for the estimation of the target cost.

21.3.3 Concatenation Cost

The ideal concatenation cost should reflect the perceived discontinuity between successive units, so as to favor the

selection of units such that *cut-and-paste* concatenation will not be noticed.

Many proposals have been made for the best definition of the concatenation cost $C^{c}(u_{i-1}, u_i)$, based on a parameterization (i. e., using some speech model) of the initial and final speech frames of concatenated units, combined with subcosts associated with the values of these q parameters:

$$C^{c}(u_{i-1}, u_i) = \sum_{j=1}^{q} w_j^{c} C_j^{c}(u_{i-1}, u_i) \,. \qquad (21.11)$$

Among the parameters used for this purpose, the most prominent are: formant values, linear prediction (LP) spectra, line spectrum frequencies (LSFs), mel-frequency cepstrum coefficients (MFCCs), and various other frequency-modified spectral representations. Distances between the values of these multi-dimensional parameters vary from the Euclidian distance to the Kullback–Leibler divergence, through weighted Euclidian distances or the Mahalanobis distance. Although it is not clear which one leads to optimal results in all cases (i. e., for all recording conditions and all languages, if possible), several studies have compared combinations of these options on specific corpora, by computing correlations between the values of these parameter/distance combinations and the impression of discontinuity reported by human listeners [21.28]. Clearly, this question is still open; its solution obviously requires (and will lead to) better modeling of perceived discontinuities in speech.

A particular case arises if the complete target sentence, or a significant part of it, is available in the corpus. In this case, the corresponding consecutive units should obviously be selected. This is easily achieved by setting the concatenation cost to zero for originally consecutive units in the speech corpus. As a result, the remarkable naturalness of speech produced by unit selection therefore comes from the fact that they tend to avoid as many concatenation points as possible, by selecting the largest sequences of originally consecutive units [hence the name, nonuniform units (NUU), that is sometimes used for this technique]. The challenge is to make sure that this does not happen at the expense of intelligibility: using units with low concatenation cost but high target cost can lead to a high degree of naturalness (which at first impresses the naïve listener) but low intelligibility.

21.3.4 Speech Corpus

Like any other expression of a natural language, however, speech obeys some form of Zipf's law: *In a corpus of natural language utterances, the frequency of any word is roughly inversely proportional to its rank in the frequency table.* The resulting power-law probability distribution implies that speech is composed of a *large number of rare events* (LNRE) [21.29]. It was estimated, for instance, that in order for a diphone database to cover a randomly selected sentence in English with a probability of 0.75 (meaning that the probability is 0.75 that all the diphones required to produce the sentence are available in the database), 150,000 units are required. This corresponds to a speech corpus of about 5 h. Yet the estimation was based on a limited number of contextual features [21.30]. Most industrial systems today use 1–10 h of speech, i. e., 150–1500 Mb, and reach very high quality synthesis *most of the time*.

Unit selection is unquestionably the best option for *limited-domain* speech synthesis [21.31], in which the data sparsity problem is avoided by constraining the vocabulary and the syntactic and prosodic complexity of the sentences to be synthesized. For the synthesis of unrestricted text, a standard interim solution to data sparsity consists of enriching the initial large speech database with a list of *safety units* (typically, a complete set of diphones), aimed at completely covering the phonetics (here meaning the coarticulation) of the language, regardless of prosody. This provides the TTS system with a fallback strategy in case of data unavailability, and ensures a lower-bound naturalness similar to that of fixed-inventory systems in the worst case (which is fortunately only exceptionally encountered). Another option is to try to obtain increasingly large speech databases. The latest TTS system developed at ATR, XIMERA, is based on a 110 h corpus of a Japanese male and a 60 h corpus of a Japanese female [21.32].

Another consequence of the LNRE effect is that designing the corpus (from which the unit database is extracted) to ensure maximum coverage is of primary importance. Several greedy algorithms that progressively build a list of sentences taken from a very large text corpus so as to reach maximum coverage with a minimum number of sentences have been proposed [21.26].

Speech segmentation also plays an important part in the creation of a speech corpus. Given the size of the speech corpus, only automatic speech segmentation methods can reasonably be used, based on forced alignment of speech using HMM models. The HMM models of phonemes used in this case differ from those used in speech recognition in that they are trained from the speaker-dependent data of the target speaker. While such an automatic method inevitably produces segmentation errors (most of them due to a bad pho-

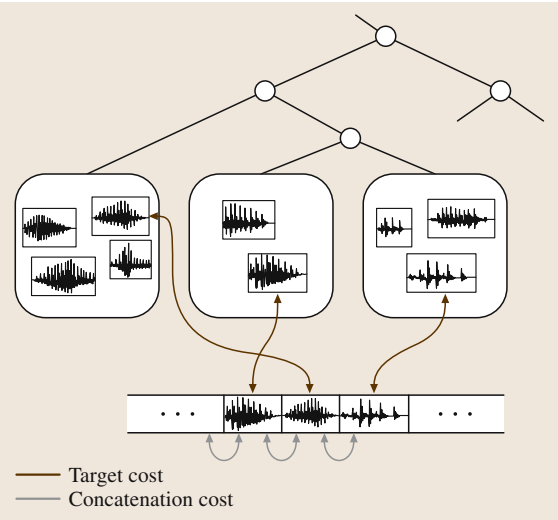


Fig. 21.10 Context-oriented clustering as a means of selecting candidate units for a given target (after [21.34])

netic transcription of the corpus), it is found that good context coverage and consistent segmentation by HMMs typically overcomes the drawback of an imperfect automatic segmentation when compared to manual segmentation [21.33].

It should be noted, however, that the development of larger segmented speech corpora has a high cost. Hence, it is becoming increasingly difficult for companies to deliver tailored voices on demand. It is also worth noticing that large unit-selection speech corpora have recently been made available for research and assessment purposes, in the framework of the Blizzard challenge [21.32]. Similarly, the voice quality offered by today's TTS systems is somehow uniform. Given the increasing quest for *expressive* speech synthesis, unit selection typically leads to parallel recordings of several speech corpora, each with an identifiable expression; this adds to the cost of a system. Chapters 24 and 25 cover these areas in detail.

Last but not least, given the size of the speech database required to implement a speech synthesizer based on unit selection, many efforts have been reported towards *pruning* the corpus, and using efficient speech *coding* algorithms. Pruning the corpus down to a given size can be done by ordering units in the original corpus as a function of their frequency of use (estimated by running the TTS on a large input text corpus), and then retaining all units up to the required corpus size. It is also useful to spot and remove outlier units in the database [21.35]. Speech coding implies the use of time-domain coders for moderate data-compression ratios (typically 1:5) or the parameterization of speech using models, which leads to much higher compression ratios at the expense of some signal degradation and additional computational cost [21.36].

21.3.5 Computational Cost

Computing target costs from each target in a large speech database to each candidate unit for that target is found to be time consuming. By recycling the COC principle mentioned in Sect. 21.2.4 and extending it to contextual *and* prosodic features (pitch, duration, and amplitude), it has been possible to drastically reduce the computational load for unit selection [21.37].

The main novelty here is that, once a COC tree is built for a given phoneme, not only is the tree retained, but also its leaves, i. e., the units that have been clustered based on their feature vectors. At run time, for each target in the sentence to synthesize, the related COC is browsed, so that only the units in the appropriate leaves are considered as candidate units (Fig. 21.10). The same idea has been used in the IBM and Microsoft TTS systems, based on [21.21]: after some preliminary segmentation of a speech corpus using HMM models, COC is applied to units associated with HMM states [21.33, 38].

The computation of concatenation costs is by far the most time-consuming operation in unit selection. Precomputing all possible concatenation costs is clearly prohibitive given the large number of units in the database. However, it was found that a limited number (typically one million) of unit pair concatenation costs provides a coverage of 99% of all concatenation costs actually computed. It is thus possible to store this subset in memory, which increases the speed of unit selection by a factor of four [21.39].

21.4 Statistical Parametric Synthesis

One of the major problems encountered in unit selection is data sparsity (see the discussion in Sect. 21.3.4). Synthetic utterances can be perfectly natural if the target sequence of units happens to be available in a continu-

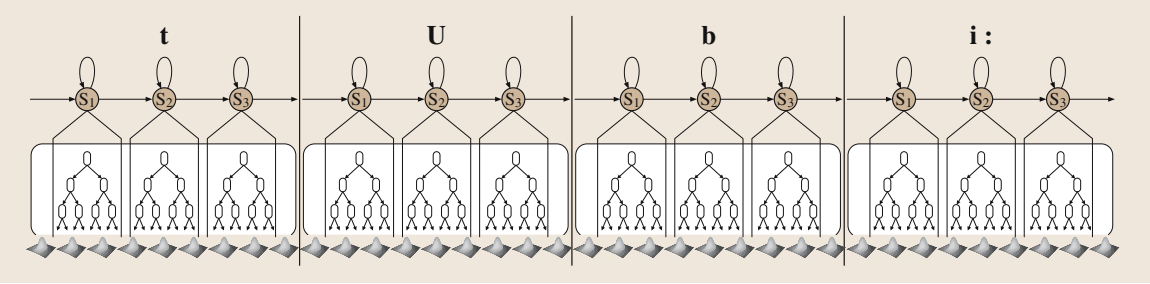


Fig. 21.11 A statistical parametric model for the spectral dynamics of the words 'to be' (after [21.34])

ous sentence in the database, but they can also embody disturbing discontinuities (which can hamper intelligibility) when the required targets are not available. In a word, unit selection lacks the capacity to generalize to unseen data.

In contrast, ASR techniques based on HMM/GMM models have developed such a generalization property, by making use of context-oriented distribution clustering for parameter tying (Chap. 27). This has motivated several research teams to make a very promising step, by considering the speech synthesis problem as a statistical analysis/sampling problem. Actually, this idea arose as early as 1989 [21.41], although the early papers on the subject only reported poor synthetic speech quality (which only adds to the merit of the researchers who pursued in this area).

In the so-called *HMM-based speech synthesis* approach, implemented in [21.34], speech is described as the acoustic realization of a sequence of phonemes. Each phoneme is modeled by a three-state HMM. Each state emits spectral feature vectors (typically MFCCs) with emission probabilities given by multivariate Gaussians associated with the leaves of a context clustering tree (Fig. 21.11). This model is very much inspired by similar work in ASR. Additionally, the pitch and duration of phonemes are modeled by separate clustering trees and Gaussians (Fig. 21.11).

The new idea in HMM-based synthesis, whose reach is certainly not limited to speech synthesis, is to use this model to generate a stream of parameters (related to a given parametric model of speech), which can then be converted into speech (Fig. 21.12).

21.4.1 HMM-Based Synthesis Framework

In the following paragraphs, we will assume that the models of Fig. 21.13 have previously been trained [21.34] so as to maximize the probability of the speech corpus given the model. The training part is similar to that of an ASR system, except the contextual factors that are used when building the context clustering trees incorporate phonetic, stress, and syntactic information (which is *produced* by the TTS system, while it is not accessible to an ASR system).

From the target sequence of phonemes (augmented with stress marks and syntactic information), a sentence HMM λ is constructed (again, as in Fig. 21.11) by concatenating context-dependent phoneme HMMs. The state durations of the phoneme HMMs are determined so as to maximize the output probability of state durations given their phonetic and syntactic context. A sequence of spectral parameters $\boldsymbol{o} = [\boldsymbol{o}_1^{\mathrm{T}}, \ldots, \boldsymbol{o}_t^{\mathrm{T}}, \ldots, \boldsymbol{o}_T^{\mathrm{T}}]^{\mathrm{T}}$ is then determined in such a way that the output probability of this sequence given the HMM model is maximized, where T is the number of frames in the observation vector sequence (known from the previously estimated state durations).

More precisely, since the duration of each HMM state is known, the sequence of spectral vectors $\boldsymbol{o}_t^{\mathrm{T}}$ to be produced is associated with a given sentence HMM state sequence $\boldsymbol{q} = \{q_1, \ldots, q_t, \ldots, q_T\}$. For each such HMM state, the contextual information of the related phoneme is used to find which leaf of the COC tree (associated with this phoneme) will model the production of the state-dependent spectral parameters. Since the

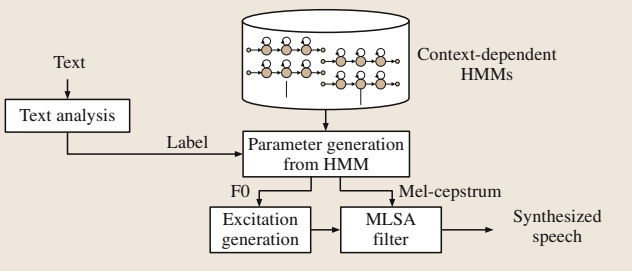


Fig. 21.12 Speech synthesis using a statistical parametric model (after [21.40])

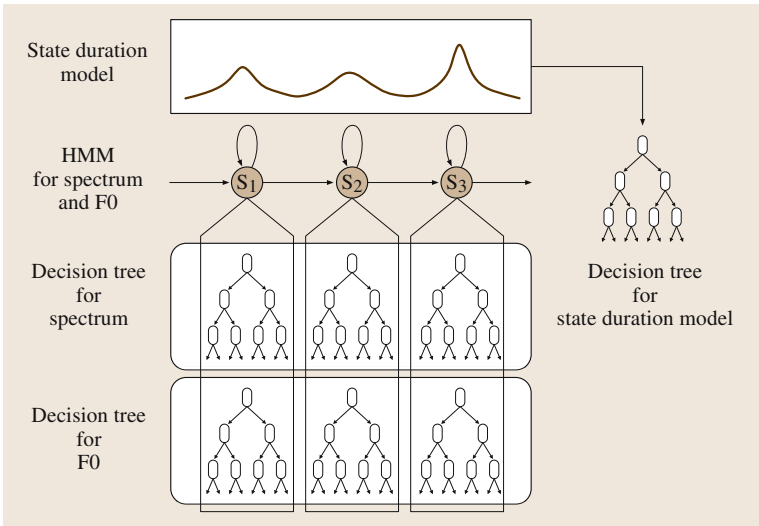


Fig. 21.13 A statistical parametric model for the spectrum F0 and duration of a phoneme (after [21.34])

spectral features of the speech segments in each leaf are described by a single multivariate Gaussian distribution $N(\boldsymbol{\mu}_{q_t}, \boldsymbol{\Sigma}_{q_t})$, the $P(\boldsymbol{o}|\boldsymbol{q}, \lambda)$ is given by

$$P(\boldsymbol{o}|\boldsymbol{q}, \lambda) = \prod_{t=1}^{\mathrm{T}} N(\boldsymbol{o}_t|\boldsymbol{\mu}_{q_t}, \boldsymbol{\Sigma}_{q_t}) \,. \tag{21.12}$$

Stated as such, however, the sequence of spectral parameters produced by each state of the model will obviously be composed of a sequence of identical spectral parameters: the mean of the underlying multivariate Gaussian (which by nature maximizes the Gaussian distribution). The main feature of HMM-based synthesis is thus the use of *dynamic* features, through the inclusion of dynamic coefficients in the feature vector.

This too, is typical of HMM-based ASR techniques (which make intensive use of the so-called delta and delta-delta features; Chap. 27), but poses a specific problem in HMM-based speech synthesis: how to find a sequence of (static *and* dynamic) spectral features which at the same time has maximum likelihood given the HMM model *and* the dynamic constraints. This is explored in the next paragraphs, widely inspired from [21.40].

It is assumed that the speech parameter vector $\boldsymbol{o}_t$ consists of the M-dimensional static feature vector

$$\boldsymbol{c}_t = [c_t(1), c_t(2), \ldots, c_t(M)]^{\mathrm{T}} \tag{21.13}$$

and of the $1, \ldots, (D-1)$-th order dynamic feature vectors

$$\boldsymbol{o}_t = \left[\boldsymbol{c}_t^{\mathrm{T}}, \boldsymbol{\Delta}^{(1)}\boldsymbol{c}_t^{\mathrm{T}}, \ldots, \boldsymbol{\Delta}^{(D-1)}\boldsymbol{c}_t^{\mathrm{T}}\right]^{\mathrm{T}} , \tag{21.14}$$

where $\boldsymbol{\Delta}^{(d)}\boldsymbol{c}_t$ is the d-th dynamic feature vector given by

$$\boldsymbol{\Delta}^{(d)}\boldsymbol{c}_t = \sum_{\tau=-L_-^{(d)}}^{L_+^{(d)}} w^{(d)}(\tau)\boldsymbol{c}_{t+\tau} \tag{21.15}$$

and $w^{(d)}(\tau)$ is the window coefficient used for calculating the d-th dynamic feature.

Equation (21.12) can be rewritten in matrix form as

$$\boldsymbol{o} = \boldsymbol{W}\boldsymbol{c} \,, \tag{21.16}$$

where

$$\begin{aligned}
\boldsymbol{c} &= \left[\boldsymbol{c}_1^{\mathrm{T}}, \boldsymbol{c}_2^{\mathrm{T}}, \ldots, \boldsymbol{c}_T^{\mathrm{T}}\right]^{\mathrm{T}} , \\
\boldsymbol{W} &= [\boldsymbol{W}_1, \boldsymbol{W}_2, \ldots, \boldsymbol{W}_T]^{\mathrm{T}} \otimes \boldsymbol{I}_{M\times M} , \\
\boldsymbol{W}_t &= \left[\boldsymbol{w}_t^{(0)}, \boldsymbol{w}_t^{(1)}, \ldots, \boldsymbol{w}_t^{(D-1)}\right] , \\
\boldsymbol{w}_t^{(d)} &= \left[0, \ldots, 0, w^{(d)}(-L_-^{(d)}), \ldots, w^{(d)}(L_+^{(d)}),\right. \\
&\qquad \left. 0, \ldots, 0\right]^{\mathrm{T}}
\end{aligned} \tag{21.17}$$

in which $L_-^{(0)} = L_+^{(0)} = 0$ and $w^{(0)}(0) = 1$.

Maximizing (21.12) can thus be rewritten to take the dynamic constraints into account, i. e., by maximizing $P(\boldsymbol{W}\boldsymbol{c}|\boldsymbol{q}, \lambda)$ or its logarithm:

$$\frac{\partial \log P(\boldsymbol{W}\boldsymbol{c}|q, \lambda)}{\partial \boldsymbol{c}} = 0 \,. \tag{21.18}$$

Notice that (21.12) can also be rewritten as

$$P(\boldsymbol{o}|q, \lambda) = N(\boldsymbol{o}|\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q) \tag{21.19}$$

by considering each multivariate Gaussian $N(\boldsymbol{\mu}_{q_t}, \boldsymbol{\Sigma}_{q_t})$ as a component of a (yet higher dimension) multivariate Gaussian $N(\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$.

Equation (21.18) then produces a closed-form solution

$$\boldsymbol{R}_q \boldsymbol{c} = \boldsymbol{r}_q \,, \tag{21.20}$$

with:

$$\begin{aligned} \boldsymbol{R}_q &= \boldsymbol{W}^{\mathrm{T}} \boldsymbol{\Sigma}_q^{-1} \boldsymbol{W} \,, \\ \boldsymbol{r}_q &= \boldsymbol{W}^{\mathrm{T}} \boldsymbol{\Sigma}_q^{-1} \boldsymbol{\mu}_q \,. \end{aligned} \tag{21.21}$$

This remarkable solution (21.20) produces the (static) vector sequence $\boldsymbol{c}$ maximizing (21.19) under the constraints (21.16).

21.4.2 The State of the Art and Perspectives

HMM-based speech synthesis currently produces speech with remarkable fluidity (smoothness), but rather poor voice quality. The resulting buzziness is still clearly due to the use of a source–filter model based on linear prediction.

It has, however, several important potential advantages over unit selection. First, its use of context clustering is far more flexible than that of unit selection, since it allows for the creation of separate trees for spectral parameters F_0 and duration. Second, its coverage of the acoustic space is better, given the generative capability of the HMM/COC/Gaussian models. Even more importantly, it embodies a complete model of natural speech with very limited footprint (1 MB). Last but not least, it provides a natural framework for voice modification and conversion.

Recently, statistical parametric synthesis has been adapted to the HMM-based synthesis of streams of articulatory parameters, instead of vocal-tract spectrum parameters [21.42]. In this case, a mapping function is required from the articulatory to acoustic domains. This attempt to unify statistical and articulatory approaches is very promising.

Another possible development of HMM-based synthesis is to use its output as target acoustic features for unit selection, as in XIMERA [21.43].

21.5 Conclusion

Clearly, the winning paradigm today in the industry is unit selection, while more-rudimentary diphone-based systems are still used for low-cost consumer products. Statistical parametric synthesis offers very promising results, and allows a degree of flexibility in speech control that is not possible with unit selection.

Some speech synthesizers are also available for free for research purposes. See for instance the MBROLA (http://tcts.fpms.ac.be/synthesis/mbrola.html), FESTIVAL (http://www.cstr.ed.ac.uk/projects/festival/), FestVox (http://festvox.org/) or FreeTTS (http://freetts.sourceforge.net/docs/) projects. See also TTSBOX, the Matlab tutorial toolbox for text-to-speech synthesis (http://tcts.fpms.ac.be/projects/ttsbox/).

References

21.1 C.M. Harris: A study of the building blocks in speech, J. Acoust. Soc. Am. **25**, 962–969 (1953)

21.2 R.V. Shannon, F.G. Zeng, V. Kamath, J. Wygonski, M. Ekelid: Speech recognition with primarily temporal cues, Science **13**(5234), 270 (1995)

21.3 N.R. Dixon, H.D. Maxey: Terminal analog synthesis of continuous speech using the diphone method of segment assembly, IEEE Trans. ASSP **AU-16**(1), 40–50 (1968)

21.4 T. Dutoit: *An Introduction to Text-To-Speech Synthesis* (Kluwer Academic, Dordrecht 1997)

21.5 B. Bozkurt, T. Dutoit, R. Prudon, C. d'Alessandro, V. Pagel: Improving the quality of MBROLA synthesis for non-uniform units synthesis. In: *Text to Speech Synthesis: New Paradigms and Advances*, ed. by S. Narayanan, A. Alwan (Prentice-Hall, Upper Saddle River 2004)

21.6 E. Moulines, F. Charpentier: Pitch synchronous waveform processing techniques for text-to-speech synthesis using diphones, Speech Commun. **9**, 5–6 (1990)

21.7 Y. Stylianou: Applying the harmonic plus noise model in concatenative synthesis, IEEE Trans. Speech Audio Process. **9**(1), 21–29 (2001)

21.8 W. Verhelst, M. Roelands: An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech, Proc. ICASSP 93, Vol. II (1993) pp. 554–557

21.9 N. Morita, F. Itakura: Time-scale modification algorithm for speech by use of pointer interval control overlap and add (PICOLA) and its evaluation, Proc. Annu. Meeting of Acoust. Soc. Jpn., Vol. 86 (1986) pp. 9–16

21.10 R.J. Mac Aulay, T.F. Quatieri: Speech analysis/synthesis based on a sinusoidal representation, IEEE Trans. Acoust. Speech Signal Process. **34**, 744–754 (1986)

21.11 J. Marques, L. Almeida: Frequency-varying sinusoidal modeling of speech, IEEE Trans. Acoust. Speech Signal Process. **37**(5), 763–765 (1989)

21.12 M.W. Macon: *Speech Synthesis Based on Sinusoidal Modeling*, Ph.D. Dissertation (Georgia Institute of Technology, Atlanta 1996)

21.13 T. Dutoit, H. Leich: MBR-PSOLA: Text-to-speech synthesis based on an MBE resynthesis of the segments database, Speech Commun. **13**, 435–440 (1993)

21.14 A. Conkie, S. Isard: Optimal coupling of diphones, Proc. 2nd ESCA/IEEE Workshop On Speech Synthesis Mohonk, ed. by J. Olive (1994)

21.15 J. Wouters, M.W. Macon: Control of spectral dynamics in concatenative speech synthesis, IEEE Trans. Speech Audio Process. **9**(1), 30–38 (2001)

21.16 V. Aubergé: *La synthèse de la parole: des règles aux lexiques*, Ph.D. Dissertation (Institut de la Communication Parlée, Grenoble 1991), in French

21.17 D. Bigorne, O. Boeffard, B. Cherbonnel, F. Emerard, D. Larreur, J.L. Le Saint-Milon, I. Metayer, C. Sorin, S. White: Multilingual PSOLA text-to-speech system, Proc. Int. Conf. Acoust. Speech Signal Process., Vol. 2 (1993) pp. 187–190

21.18 T. Portele, W. Sendlemeier, W. Hess: HADIFIX, a system for German speech synthesis based on demisyllables, diphones, and suffixes, Proc. First ESCA Workshop on Speech Synthesis (1990) pp. 161–164

21.19 S. Nakajima: Automatic synthesis unit generation for English speech synthesis based on multi-layered context oriented clustering, Speech Commun. **14**, 313–324 (1994)

21.20 R. Donovan, P. Woodland: Improvements in an HMM-based speech synthesizer, Proc. Eurospeech 95, Vol. 1 (1995) pp. 573–576

21.21 J.P. Olive: Rule synthesis of speech from diadic units, Proc. ICASSP, Vol. 77 (1977) pp. 568–570

21.22 Y. Sagisaka, N. Kaiki, N. Iwahashi, K. Mimura: ATR ν-TALK speech synthesis system, Proc. ICSLP 92, Vol. 1 (1992) pp. 483–486

21.23 A.J. Hunt, A.W. Black: Unit selection in a concatenative speech synthesis system using a large speech database, Proc. Int. Conf. Acoust. Speech Signal Process. (ICASSP '96), Vol. 1 (1996) pp. 373–376

21.24 N. Campbell, A. Black: Prosody and the selection of source units for concatenative synthesis. In: *Progress in Speech Synthesis*, ed. by J. van Santen, R. Sproat, J. Olive, J. Hirshberg (Springer, Berlin, Heidelberg 1995)

21.25 M. Beutnagel, A. Conkie, J. Schroeter, Y. Stylianou, A. Syrdal: The AT&T next-gen TTS system, Proc. Joint Meeting of ASA (1999)

21.26 M. Balestri, A. Paechiotti, S. Quazza, P.L. Salza, S. Sandri: Choose the best to modify the least: a new generation concatenative synthesis system, Proc. Eurospeech, Vol. 99 (1999) pp. 2291–2294

21.27 P. Taylor, A.W. Black: Speech synthesis by phonological structure matching, Proc. Eurospeech, Vol. 99 (1999) pp. 623–626

21.28 J. Vepa, S. King: Join cost for unit selection speech synthesis. In: *Speech Synthesis*, ed. by A. Alwan, S. Narayanan (Prentice-Hall, Upper Saddle River 2004)

21.29 B. Möbius: Rare events and closed domains: Two delicate concepts in speech synthesis, Int. J. Speech Technol. **6**(1), 57–71 (2003)

21.30 J.P.H. van Santen: Combinatorial issues in text-to-speech synthesis, Proc. Euro. Conf. Speech Commun. Technol., Vol. 5 (1997) pp. 2511–2514

21.31 A. Black, K. Lenzo: Limited domain synthesis, Proc. ICSLP (2000) pp. 411–414

21.32 C. Bennett, A. Black: The Blizzard Challenge 2006, Proc. Blizzard Challenge 2006 (2006)

21.33 X. Huang, A. Acero, J. Adcock, H. Hon, J. Goldsmith, J. Liu, M. Plumpe: Whistler: A trainable text-to-speech system, Proc. ICSLP, Vol. 96 (1996) pp. 659–662

21.34 K. Tokuda, H. Zen, A. Black: An HMM-based approach to multilingual speech synthesis. In: *Text to Speech Synthesis: New Paradigms and Advances*, ed. by S. Narayanan, A. Alwan (Prentice Hall, Upper Saddle River 2004) pp. 135–153

21.35 Y. Zhao, M. Chu, H. Peng, E. Chang: Custom-tailoring TTS voice font – keeping the naturalness when reducing database size, Proc. Eurospeech, Vol. 2003 (2003) pp. 2957–2960

21.36 D. Chazan, R. Hoory, Z. Kons, A. Sagi, S. Shechtman, A. Sorin: Small footprint concatenative text-to-speech synthesis system using complex spectral envelope modeling, Proc. Interspeech, Vol. 2005 (2005) pp. 2569–2572

21.37 A.W. Black, P. Taylor: Automatically clustering similar units for unit selection in speech synthesis, Proc. Eurospeech, Vol. 2 (1997)

21.38 R.E. Donovan, E.M. Eide: The IBM trainable speech synthesis system, Proc. Int. Conf. Spoken Lang. Process., Vol. 5 (1998) pp. 1703–1706

21.39 M. Beutnagel, M. Mohri, M. Riley: Rapid unit selection from a large speech corpus for concatenative speech synthesis, Proc. Eurospeech '99, Vol. 2 (1999) pp. 607–610

21.40 H. Zen, K. Tokuda, T. Kitamura: An introduction of trajectory model into hmm-based speech synthesis, Proc. Speech Synthesis Workshop (2005)

21.41 A. Falaschi, M. Giustiniani, M. Verola: A hidden Markov model approach to speech synthesis, Proc. Eurospeech, Vol. 1989 (1989) pp. 2187–2190

21.42 K. Nakamura, T. Toda, Y. Nankaku, K. Tokuda: On the use of phonetic information for mapping from articulatory movements to vocal tract spectrum, Proc. ICASSP, Vol. 06 (2006) pp. 93–96

21.43 H. Kawai, T. Toda, J. Ni, M. Tsuzaki, K. Tokuda: XIMERA: A new TTS from ATR based on corpus-based technologies, Proc. 5th ISCA Speech Synthesis Workshop (2004) pp. 179–184

25. Expressive/Affective Speech Synthesis

N. Campbell

The focus of speech synthesis research has recently shifted from *read speech* towards more *conversational* styles of speech, in order to reproduce those situations where a speech synthesis is used as part of a dialogue. When a speech synthesizer is used to represent the voice of a cognisant agent, whether human or simulated, there is need for more than just the intelligible portrayal of linguistic information; there is also a need for the expression of affect. This chapter reviews some recent advances in the synthesis of expressive speech and shows how the technology can be adapted to include the display of affect in conversational speech.

The chapter discusses how the presence of an interactive and active partner in a conversation can greatly affect the styles of human speech and presents a model of the cognitive processes that result in these differences, which concern not just the acoustic prosody and phonation quality of an utterance, but also its lexical selection and phrasing. It proposes a measure of the ratio of paralinguistic to linguistic content in an utterance as a means of quantifying the expressivity of a speaking style, and closes with a description of a phrase-level concatenative speech synthesis system that is currently in development.

25.1 Overview

Expressive speech synthesis, or the synthesis of affective speech is a relatively new area of research, prompted by a shift in the technology from *reading machines* towards *talking machines* [25.1]. This arose from the need for machines to be able to express more than just the phonetic content of an utterance and to be able to make use of more human-like subtlety in the generation of its prosodic characteristics. Accordingly, speech synthesis research has shifted from *read speech* towards more *conversational* styles of speech. For those situations where speech synthesis is used as part of a dialogue, i. e., where it represents the voice of a cognisant agent, whether human or simulated, then there is a need for more than just the intelligible portrayal of linguistic information; there is also a need for the expression of affect.

Not to be confused with the simpler term 'emotion', 'affect' as defined in this context refers to the human characteristics that govern the various subtleties of intonation and phrasing, which reveal extralinguistic and paralinguistic information about the speaker, about the speaker's relationships with the hearer, and about the progress of the discourse and the degrees of mutual understanding attained throughout its progress.

Expressive speech is that which reveals affect. The challenge in synthesizing expressive speech lies in the adequate specification of affective states in the input and in the manipulation or realization of prosodic characteristics to express them in the output speech. This chapter will start by describing the types of speech variation that will be necessary to synthesize, discuss

some techniques for eliciting them, and then close by describing some of the approaches that have been taken to tackle the problems of their modeling and realization.

25.2 Characteristics of Affective Speech

Speech per se is a meaningful type of noise which can take place with or without a hearer, but the nature of interactive speech specifically depends on the presence and purposes of a hearer. Broadcast speech represents one extreme of a hypothetical continuum of listener involvement, with romantic murmurings or infant-directed speech perhaps forming the other; currently only the former can be well modeled for synthesis, although there are calls for a more-personal and intimate form of speaking in many current applications of speech synthesis.

Broadcast speech assumes a general audience that is perhaps interested but not necessarily present, and certainly unable to provide any immediate relevant feedback. The focus here is on a unidirectional transmission of information. Conversational speech on the other hand typically requires an active and participating listener. The focus here is on the interaction. The listener may be remote, as in the case of telephone speech, but is required to give constant and timely feedback. The nature of the conversational speech changes according to these feedback reactions from the listener. Classroom speech or lectures can be considered intermediate on this continuum, since the audience is distant but present, able to show comprehension or otherwise, allowing the speaker to moderate and modify the content and speaking style accordingly.

To summarize, affective speech differs from traditional read speech in depending on a model of the listener being incorporated into the speech production process. The task of expressive speech synthesis is first to represent the higher-level interpersonal structures that motivate the speech [25.2], and then to model their lower-level consequences through prosodic control of the output speech signal.

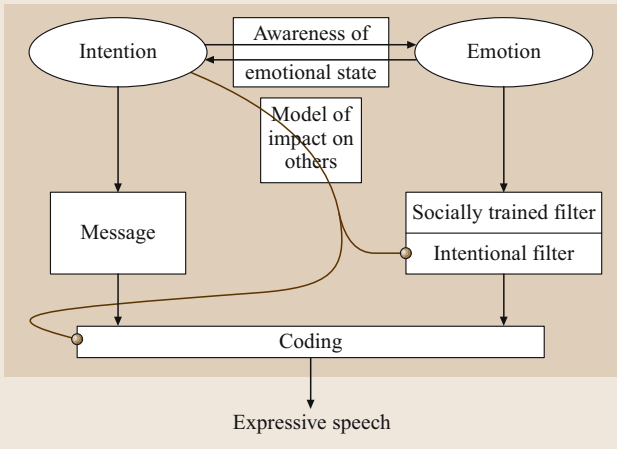


Fig. 25.1 A proposed model to explain the interaction of affective and intentional information in the generation of a speech utterance. The ovals represent hidden processes or states that are not subject to conscious control but which serve as driving forces behind the production of the utterance. These are substantiated in the form of a message and filters, with constraints that are subject to a model of the potential impact on the listener, that determine the muscular coding for the production of the utterance with its resulting prosody and phonetic sequence

25.2.1 Intentions and Emotions

A representation of the cognitive processes involved in determining a specific speaking style for expressing affect-related information is given in Fig. 25.1, which schematizes the flow of activity that results in expressive speech. The interaction of emotions and intentions in the generation of an expressive speech utterance forms the higher-level part of this framework. In the figure, both *emotions* and *intentions* are represented by ovals, signifying their intangibility; i. e., that they act as a force but are not readily available for identification, control, or conscious appraisal.

The model posits these two underlying or hidden cognitive forces, intention and emotion, as drivers that provide the motivation for a basic communicative event that is to be realized in the form of an utterance in a discourse. They are distinguished from the more-tangible message and filters to be discussed in more detail below. These in turn govern a coding level of processing, which produces lower-level commands for the muscles that produce the speech and accompanying facial or bodily gestures.

A given combination of intentions and emotions represents an underlying sociopsychological state within the speaker, which is raw and unbound by linguistic or

social conventions. There may of course be some interaction between emotions and intentions, since intentions can be triggered by emotions, and emotions can in turn be subdued or amplified intentionally as part of a rational process, such as when a speaker forces herself to smile so that her voice will become happier. These are abstract causes that underly the more-tangible level of message and filters that particularly concerns us here, but they result in subtle differences in force of expression and need to be modeled carefully, since human listeners are particularly sensitive to their variations.

25.2.2 Message and Filters

The message gives form to the underlying intentions and constitutes a speech act, a discourse act, and a social event. It is not yet a text. It may be a greeting, a complaint, provision of information, request for information, etc., and may stand alone or function as a dependent element of a larger discourse. In many cases it will be as much phatic as informational in intent.

It is at the level of the message that the utterance begins to take shape, but its linguistic content and prosodic realization remain indeterminate at this level. For example, a greeting could take the form of 'Good morning', or 'Hi!', depending on who is being addressed, on the mood of the speaker, and on the contexts of the discourse (both social and environmental). These details are determined by the settings of the filters and realized at the level of the coding.

These filters are socially trainable. They depend to a large extent on language-specific, culture-specific and subculture-specific aspects, but can have an intentional element. They incorporate such modifiers as politeness constraints and serve to signal attitudinal relationships and interpersonal stances. The filters are shown as bilevel; depending both on social conditioning (above) and a degree of intentional control (below). It is at this lower level that the speaker takes into consideration the potential impact of an utterance on the listener [as illustrated (not coincidentally) in the center of the figure].

Whereas certain constraints may be ingrained, or determined by society and imprinted in the speaker at an early age, others are more open to conscious selection. For example, while young infants may readily and directly express whatever emotions they currently feel, older children and adults become more reserved, often concealing their true feelings or masking them for social reasons. A salesperson may wish to portray the proper company image, hiding particular strengths or weaknesses, or a call-center operator may be required to sound cheerful, even though the displayed emotion may be in conflict with that actually felt by the speaker at the time. This dichotomy provides part of the richness of spoken language and is surely parsed by the listener as part of (or alongside) the message.

Both filter levels function to control

1. what is displayed
2. what is concealed in the production of an utterance

They have an effect not just on the selection of lexical items and phrasing, but also on voice quality and prosodic aspects of phonation so that the utterance can be parsed appropriately as expressing the speaker's intentions subject to the prevailing social and psychological states and conditions. This requires a level of markup on the input far beyond that which is currently specified by the existing W3 speech synthesis markup language (SSML) conventions [25.3].

The sophistication of interactive human speech is a direct result of this multiplicity of cues that taken together contribute the intended interpretation of its linguistic and pragmatic content. The prosody and voice quality of expressive speech also encode subtle information related to the speaker's basic internal emotional states but masked by a more-superficial layer of information related to discourse goals and intentions, and social conventions governing politeness and sincerity of expression. The task of expressive speech processing, whether for synthesis or recognition, is to model the interrelationship of each of these sometimes conflicting sources of information.

25.2.3 Coding and Expression

In this model of expressive speech generation, the determination of lexical items, utterance complexity and length, phrasing, speaking rate and style, etc., is considered as taking place at the lowest level of utterance production, in a constraint-based way, subject to the higher-level constraints described above and illustrated in the main part of the figure.

Because there are usually several different ways of phrasing a proposition or eliciting backchannel information, the choice of a particular variant reveals much about the intentions and affective states of the speaker and about the contexts of the discourse. Both the text of the message and its prosodic encoding are constrained by the intentions of the speaker, subject to variations in emotional state and social as well as intentional constraints on utterance production.

Figure 25.1 shows the coding level, which ultimately produces sequences of muscular movements, to be fed by two streams of complementary information, as shown by the left and right vertical arrows; both are subject to a model of the impact of the utterance on the listener and others. This information can be similarly decoded to reveal not just the linguistic content, but also information about the speaker and the settings of the various filters.

25.3 The Communicative Functionality of Speech

The model implies that any given speech utterance contains information related not just to its propositional content, if any, but also to speaker-related and affective information, to speaker–hearer relationships, and to environmental factors etc.; i. e., in addition to the lexical content or word sequence, an utterance provides both linguistic prosody and social prosodic information [25.2, 4]. The former has a long history of being modeled in speech synthesis being predictable largely from the text (see, for example, [25.5–8]), but the latter remains a subject for continuing and intense research, being closely related to database design and corpus collection issues in the case of concatenative synthesis techniques [25.9–11].

These prosodic elements, along with related differences in voice quality, can be decoded by the listener to reveal the affective and interpersonal information that signals the speaker's position relative to the utterance and the discourse, and enables the listener to parse its intended pragmatic meaning from among the many possible linguistic candidate interpretations of the text string. The structure of conversational speech utterances (often called ill-formed) differs considerably from that of their written equivalents not because people are simple and unable to think efficiently in real time, but because speech as a medium has the benefits of multiple layers of prosodic information from which the underlying meanings and intentions of an utterance can be made clear.

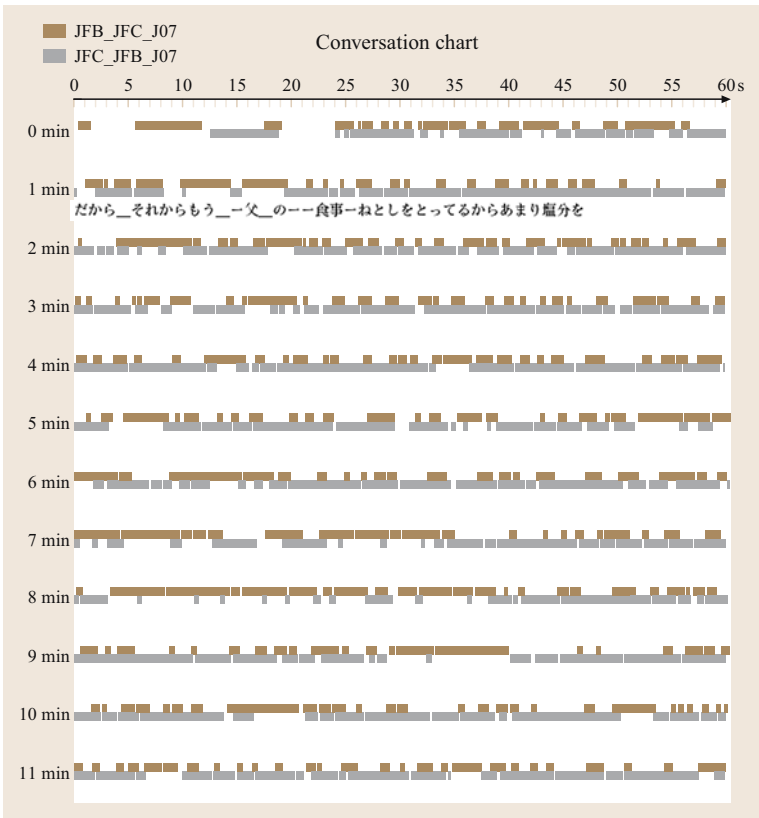


Fig. 25.2 Time-aligned speech activity in a natural telephone conversation, showing how the dominant role is passed from one participant to the other in a not very clear-cut way. The lower speaker (grey) appears to be dominant, but there is considerable overlap as they negotiate their way through the conversation

Figure 25.2 shows the interplay of speaker and listener in a short telephone conversation taken from the expressive speech processing (ESP) corpus [25.12]. It is clear that the roles are often reversed, and that even in listener mode there is considerable speech activity. These backchannel feedback signals provide the speaker with cues to the listener's cognitive state, and comprehension of and agreement with the flow of the conversation as the two mutually develop a joint understanding, or meeting of the minds [25.13].

Conversational speech utterances are not just typically shorter than their written equivalents, they often contain idiomatic phrases, phatic elements, laughs, etc., that illustrate the discourse. Being so frequent and repetitive, these inessential speech noises allow the listener, even one who is unfamiliar with the speaker, to make judgements about the speaker's affective states and discourse intentions.

Whereas the true intentions and emotions of the speaker must remain hidden, much can be inferred about them from the combination of information in the message and in the choice of speaking style (i. e., from the visible effects of the inferred filters), the listener thereby has access not just to the text of the utterance, but also to:

1. intended meaning(s)
2. speaker state(s)
3. listener status and relationship(s) to the speaker.

This is what is now being covered in the developing studies of social prosody and what is yet to be modeled in expressive speech synthesis.

25.3.1 Multiple Layers of Prosodic Information

Since its original design as a reading machine, speech synthesis research has been focussed on the conversion of written words into speech sounds, using grapheme-to-phoneme conversion, prosody prediction, and waveform generation as its three main subprocesses.

Accordingly, there is a considerable body of prosody-related research in the field of speech synthesis, but almost all of it (with very few exceptions) is related to the forms of prosody that can be predicted from the text alone. The exceptions largely concern gender-related prosodic differences, or linguistic focus. Existing speech synthesis markup languages (e.g., [25.3]) provide for the modification of prosody, but only at the lowest level of mean pitch, phoneme duration, and amplitude. Little work has yet been done on the annotation of text for the expression of the types of affective information described above.

Traditional prosody for speech synthesis is directly related to the text structure of each utterance [25.14, 15], although nowadays it is predicted usually by use of statistical models that have been trained on speech data often unrelated to the voice or personality of the synthesis speaker (e.g., [25.16, 17]). The models provide duration, pitch, and sometimes power values for each segment of the utterance, often through an intermediate phonological representation such as ToBI [25.18, 19].

25.3.2 Text Data versus Speech Synthesis

Spoken language has been extensively studied through the use of corpora for several decades now, and the differences between the types of information that can be conveyed through written texts and those that are signalled through speech are beginning to be well understood. The different types of information that are signalled by different speaking styles are also well understood and are beginning to be modeled in speech technology applications, especially paralinguistic information, which is an essential component of speech communication that is largely carried through modulations of prosody, tone of voice, and speaking style. The more formal the speech, the more constrained the types of paralinguistic information that are conveyed.

At the formal extreme we might consider a public lecture where the speaker is (sometimes literally) talking from a script, to a large number of listeners (or even to a recording device with no listeners physically present) and has minimal feedback from, or two-way interaction with, the audience. This type of spontaneous speech is perhaps the most constrained, and most resembles text. At the most informal extreme, we might consider the mumblings of young lovers. Their conversation is largely phatic, and the words might carry little of linguistic content but are instead rich in feelings. For them, talk is almost a form of physical contact.

There are many steps along the continuum between these two hypothetical extremes of speaking-style variation, and they can be distinguished by the *ratio of paralinguistic to linguistic content*, i. e., the amount of personal information that is included in the speech. The lecture, having almost no personal information and a very high amount of propositional content will result in a very low value of this measure, while the phatic mutterings will score very high.

Speech research depends on the quality and types of the data on which it is modeled. If we are to collect data that contains sufficient examples of natural spoken interactions along the whole range of this continuum of values, then low-scoring material will prove very easy to collect, but most lovers might object strongly to the prospect of having a recording device intruding upon their privacy. Thus, by far the majority of speech corpora that have been used in previous research score very poorly on this scale and as a result the speech that they contain is not very far removed from pure text in its style and content.

25.4 Approaches to Synthesizing Expressive Speech

Expressive speech synthesis is covered under many patents, among them US patent no. 5 559 927 of Sept. 1996, for a *Computer system producing emotionally-expressive speech messages* which describes "a computer system in which the sounds of different speech messages are stored or synthesized, the system being adapted to reproduce a selected speech message and to impart emotional expressivity thereto whose character depends on the user's choice".

But what is emotional expressivity? Take as an example, the website of the Signal Processing Laboratory of the University of the Basque Country (UPV/EHU) in Bilbao [25.20], which announces expressive speech as its title, and which exemplifies what I believe to be a popular and very widespread misconception about expressive speech:

The page states that:

> *Expressive speech synthesis concerns the addition of features related to the emotional state of the (synthetic) speaker as well as the specific characteristics that identify the speaker to the neutral synthetic speech. As such, aspects as emotions (fear, anger, surprise, disgust, happiness, unhappiness . . .) and voice quality features (. . .) should be considered when synthesizing expressive speech.*

It is often taken for granted that expressiveness in speech is the direct result of changes in emotional states, and furthermore that the emotions to be expressed are limited to happiness, sadness, fear, anger, surprise, and disgust (i. e., *Ekman*'s big six [25.21]). This may be an understandable assumption but it is one that does little justice to the sophistication of human interactive communication as described above, unless we are to limit expressive speech synthesis to the realms of storytelling, cartoons, and entertainment [25.22–24].

Ekman was attempting to explain facial expression when he published his seminal work, but the voice is different from the face, and in spoken conversation, what it reveals may not be the emotion so much as the interests of the speaker. In our analysis of a very large corpus of naturally occurring everyday conversations ([25.10, 25], see below), these strong emotions accounted for less than 1% of the expressive variability in the speech. However, they account for more than 90% of the related research in speech synthesis.

25.4.1 Emotion in Expressive Speech Synthesis

The modeling of expressive speech for speech synthesis perhaps started with the masters thesis of *Janet Cahn* at MIT in 1989 on generating expression in synthesized speech [25.26]. Her work resulted in an affect editor, which modified the prosodic parameters of DecTalk, changing the default settings of pitch, timing, voice quality and articulation according to the emotional states (angry, disgusted, fearful, glad, and sad) that were selected by the user.

Her first paper nearly 20 years ago [25.27] started with the words: "Synthesized speech is readily distinguished from human speech on the basis of inappropriate intonation and insufficient expressiveness", of which perhaps now only the latter part is still true, and she continued: "In synthesized speech intonation makes the message easier to understand; enhanced expressiveness contributes to dramatic effect, making the message easier to listen to," which is still completely true.

It is of particular interest to note that in the 1990 journal paper that described her thesis work [25.28], the simple list of the big six emotions is expanded (Fig. 25.3, which is Fig. 2 in her paper) to include more truly affective states: afraid, angry, annoyed, disgusted, distraught, glad, indignant, mild, plaintive, pleasant, pouting, sad, and surprised, although these were not mentioned in either the thesis or in the 1989 conference paper to the same society [25.29]. This necessary extension from emotion to affect, perhaps encouraged by a sensitive reviewer, has not yet been so readily taken up by the rest of the community of speech synthesis researchers.

A recent paper on expressive speech from AT&T [25.30] for example, describes an experiment

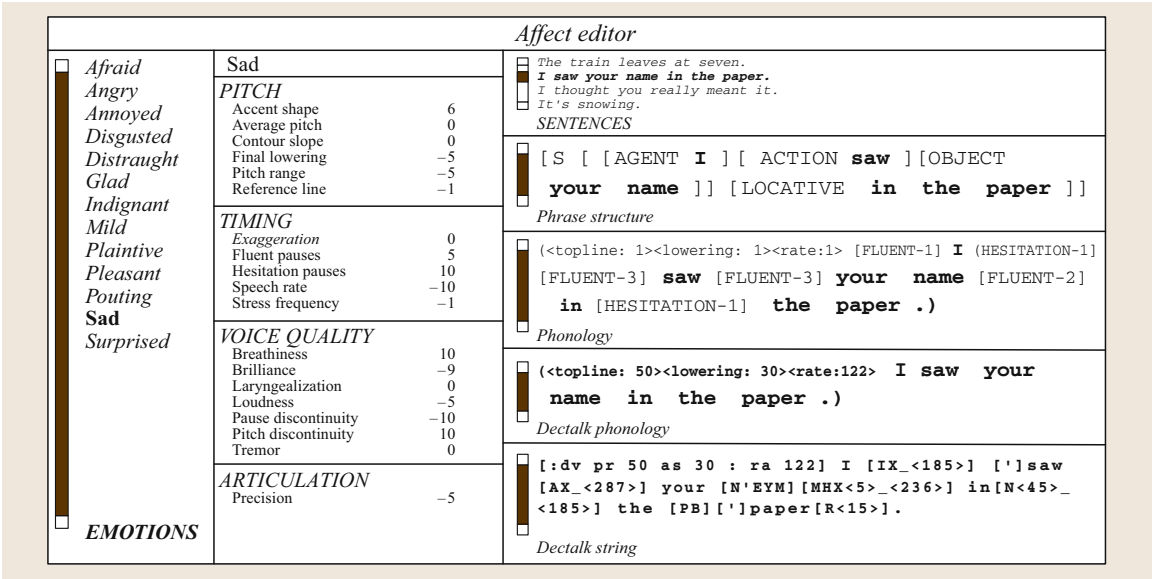


Fig. 25.3 The Effect Editor user interface. In this example, the sentence, "I saw your name in the paper." will be spoken with a Sad affect. The parameters of the model and their quantities appear in the middle column, labeled with the curent emotion (Sad). (after [25.28, Fig. 2])

in synthesizing four emotional states – anger, happiness, sadness and 'neutral' [sic] – using a concatenative speech synthesizer. The results were evaluated by conducting listening tests with 33 naive listeners and the synthesized anger was recognized with 86.1% accuracy, sadness with 89.1%, happiness with 44.2%, and neutral emotion with 81.8% accuracy.

Recent developments at MIT for embodied communicative agents are extending Cahn's research, but still primarily from the standpoint of expressing raw emotions. The Kismet project [25.31] explains:

> *emotions have a global impact on speech since they modulate the respiratory system, larynx, vocal tract, muscular system, heart rate, and blood pressure. ... when a subject is in a state of fear, anger, or joy, the sympathetic nervous system is aroused. This induces an increased heart rate, higher blood pressure, changes in depth of respiratory movements, greater sub-glottal pressure, dryness of the mouth, and occasional muscle tremor. The resulting speech is faster, louder, and more precisely enunciated with strong high frequency energy, a higher average pitch, and wider pitch range. In contrast, when a subject is tired, bored, or sad, the parasympathetic nervous system is more active. This causes a decreased heart rate, lower blood pressure, and increased salivation. The resulting speech is typically slower, lower-pitched, more slurred, and with little high frequency energy.*

Since the Kismet goals are to give the appearance of sentience to an inanimate robot, these considerations may be appropriate, but in their recognition of human speech [25.32], they acknowledge concern with the affective states and communicative intentions of the speaker, categorizing incoming speech as one of: soothing, approving, prohibiting, or neutral (i. e., robot-directed). That is, they recognize that some speech is listener-directed rather than speaker-revealing.

Part of the reason for the dominance of raw emotions in expressivity research is the ease of collecting training data. *Campbell* and *Iida* collected an hour each of happy, sad, and angry speech in addition to the standard read-speech to produce an emotional voice for CHATR [25.33, 34]. By switching databases when producing synthesized speech, listeners correctly perceived the intended emotion. However, we soon found that our users, typically speaking-impaired persons such as ALS patients, had less need to express such simple emotions and wanted instead to show interest through tone of voice, or to show concern, warmth, indifference, to be able to communicate their feelings rather than their emotional states.

IBM Research, another team that has an excellent reputation for speech synthesis, has recognized this need for listener-based expression of affect, rather than speaker-based expression of emotion [25.35]. They acknowledge the intentional basis of human communication, and state their text-to-speech research goal to be "to make synthesized speech as intelligible, natural and pleasant to listen to as human speech and have it communicate just as meaningfully". They use concatenative techniques to tune the speech to fit the text content, currently defining expressiveness as distinguishing between: a good news statement, a bad news statement, a yes/no question, and Contrastive emphasis [25.36].

The need to distinguish between urgency and calm in the voice is also realized by the automotive industry. SVOX is continuing to research and develop new ways to extend unit selection algorithms to impart various emotions and expressive voice corpora [25.37]. They claim that:

> *With SVOX SpeechCreate, expressive text-to-speech prompts can be designed and integrated into our customers system. For automotive applications, as an example, an impression of urgency can be included with driving directions that should be followed immediately. In addition, paralinguistic sounds can be added to increase the human-like quality of the speech output.*

There are many similar examples of expressive speech on the web. The interested reader is recommended to visit Felix Burk Burkhardt's site (http://emosamples.syntheticspeech.de) first of all, then to explore the Humaine links [25.38] for a review research related to the expression of emotions in speech.

25.5 Modeling Human Speech

In our own expressive speech research, we first produced a corpus. The Japan Science and Technology Corporation core research for evolutional science and technology (JST/CREST) expressive speech corpus [25.39] was collected over a period of five years by fitting a small number of volunteers with head-mounted high-quality microphones and small minidisc walkman recorders to be worn while going about their ordinary daily social interactions. Further groups of paid volunteers transcribed and annotated the speech data for a variety of characteristics, including speech-act, speaker-state, emotion, relationship to the interlocutor, etc.

Altogether, we collected 1500 h of spontaneous natural conversational speech. All the data were manually transcribed, and about 10% was further annotated. Table 25.1 shows some of the categories that were used for annotation. Speech samples can be listened to at the project website, http://feast.atr.jp/non-verbal/. We found many words and short phrases repeated frequently throughout the corpus; utterances used by the speakers more for their discourse effect than for their linguistic or propositional content. These utterances proved most difficult for the labelers to adequately categorize. They function primarily as backchannel utterances, but also serve to display a wide range of attitudinal and affective states.

We have reported elsewhere [25.4] on studies that measure the extent to which their function can be similarly perceived by different groups of listeners belonging to different cultural backgrounds and languages. In terms of quantity, more than half of the utterances in the corpus were of this predominantly nonverbal type; short words or simple syllables that occurred alone or were repeated several times in succession, often not appearing at all in a dictionary of the formal language, but forming essential components of a two-way spoken interaction.

Our labelers tested several methods of describing these conversational utterances and eventually decided on the three-level system shown in Table 25.1, whereby the following could be distinguished

1. facts about the speaker
2. facts about the utterance, and
3. separate independent evaluations could be made about the information portrayed by differences in the voice quality.

Level 1 describes the state of the speaker, and requires long-term context in order to enable a judgement. It produces an estimation of the discourse purpose of the utterance (see details below), the speaker's emotional states and mood (these labels are free input, those in the table being examples), her interest in the discourse, and finally a label to denote labeler confidence in the current decisions. The numerical labels are forced choice on a scale of high to low (see the lower part of the table) with no default or zero setting.

Level 2 describes the style of the speech, its type and purpose, as can be estimated from a short time window (i. e., with no information regarding discourse context) so that it describes the information available from lis-

Table 25.1 Three levels of labeling for describing each utterance, includingthe use of six-level forced-choice tendency scales

Level 1	**State (about the speaker)**
Purpose	A discourse-act/DA label (see text)
Emotion	Happy/sad/angry/calm
Mood	Worried/tense/frustrated/troubled/ . . .
Interest	A 6-point scale from +3 to −3, omitting 0
Confidence	A 6-point scale from +3 to −3, omitting 0
Level 2	**Style (about the speech)**
Type	Speaking-style label (open class)
Purpose	A discourse-act label (closed class)
Sincerity	Insisting/telling/feeling/recalling/acting/ . . .
Manner	Polite/rude/casual/blunt/sloppy/childish/ sexy/ . . .
Mood	Happy/sad/confident/diffident/soft/ aggressive/ . . .
Bias	Friendly/warm/jealous/sarcastic/flattering/ aloof/ . . .
Level 3	**Voice (about the sound)**
Energy	A six-point scale from +3 to −3, omitting 0
Tension	A six-point scale from +3 to −3, omitting 0
Brightness	A six-point scale from +3 to −3, omitting 0
Level 0	**Labeler**
Confidence	A six-point scale from +3 to −3, omitting 0

Six-point values	**Negative**	**Positive**
Very noticeable	−3	3
Noticeable	−2	2
Only slightly noticeable	−1	1

tening to the isolated speech utterance alone, as distinct from the same utterance situated in a discourse (i. e., we are not interested in whether the speaker is actually, e.g., angry or not, but only in whether the particular segment in question sounds angry). The *sincerity* label describes an important functional aspect of the speech, such as can be distinguished between the verbs 'insisting', 'telling', 'quoting', 'saying', 'feeling', 'recalling', 'acting', 'pretending', etc.

Manner is a bucket category that includes politeness and sexiness (which are not at all mutually contradictory) as well as childishness, sloppiness, etc. to describe the perceived attitude(s) of the speaker towards the listener. This is complemented by mood and bias, of which the former indicates the affective states of the speaker, and the latter his or her attitudes.

Level 3 describes the physical characteristics of the speaker's voice quality and speaking style in perceptual terms.

25.5.1 Discourse-Act Labeling

In order to describe the purpose or function of each utterance, a decision is first made about its directionality, which may be either offering (to the listener) or seeking (from the listener). Utterances are then broadly categorized into seven classes of discourse intentions, including: questions, opinions, objections, advice, information, greetings, and grunts. These category labels are determined by necessity as examples of each appeared in the data. As noted above, the last category accounted for almost half of the utterances in the corpus.

Under the category of 'questions', we use the following labels: WH questions, Y/N questions, repetition requests, and information requests.

Under the category of 'opinions' we use the following labels: opinion, compliment, desire, will, thanks, and apology.

Under the category of 'objections' we use the following labels: objection, complaint.

Under the category of 'advice' we use the following labels: advice, command, suggestion, offer, and inducement

Under the category of 'information' we use the following labels: give information, reading, introduce self, introduce topic, and closing

Under the category of 'greetings' we use the following labels: greeting, talking to self, asking self, checking self.

Under the category of 'grunts' we use the following labels: notice, laugh, filler, disfluency, mimic, habit, response, and backchannel. Response and backchannel utterances are further subcategorized into the following types: agree, understand, convinced, accept, interested, not convinced, uncertain, negative, repeat, self-convinced, notice, thinking, unexpected, surprise, doubt, impressed, sympathy, compassion, exclamation, listening, and other.

25.5.2 Expressive Speech and Emotion

The experience gained from this labeling process has caused us to now rethink some of our original assumptions. We started out by attempting to overcome Labov's observer's paradox on the assumption that long-term exposure to a recording device would eventually cause the wearer to become so familiar with it that it no longer becomes a hindrance to normal spoken interaction, even of a highly personal kind. This has proved to be the case, and is confirmed by the variety of speech that we have collected.

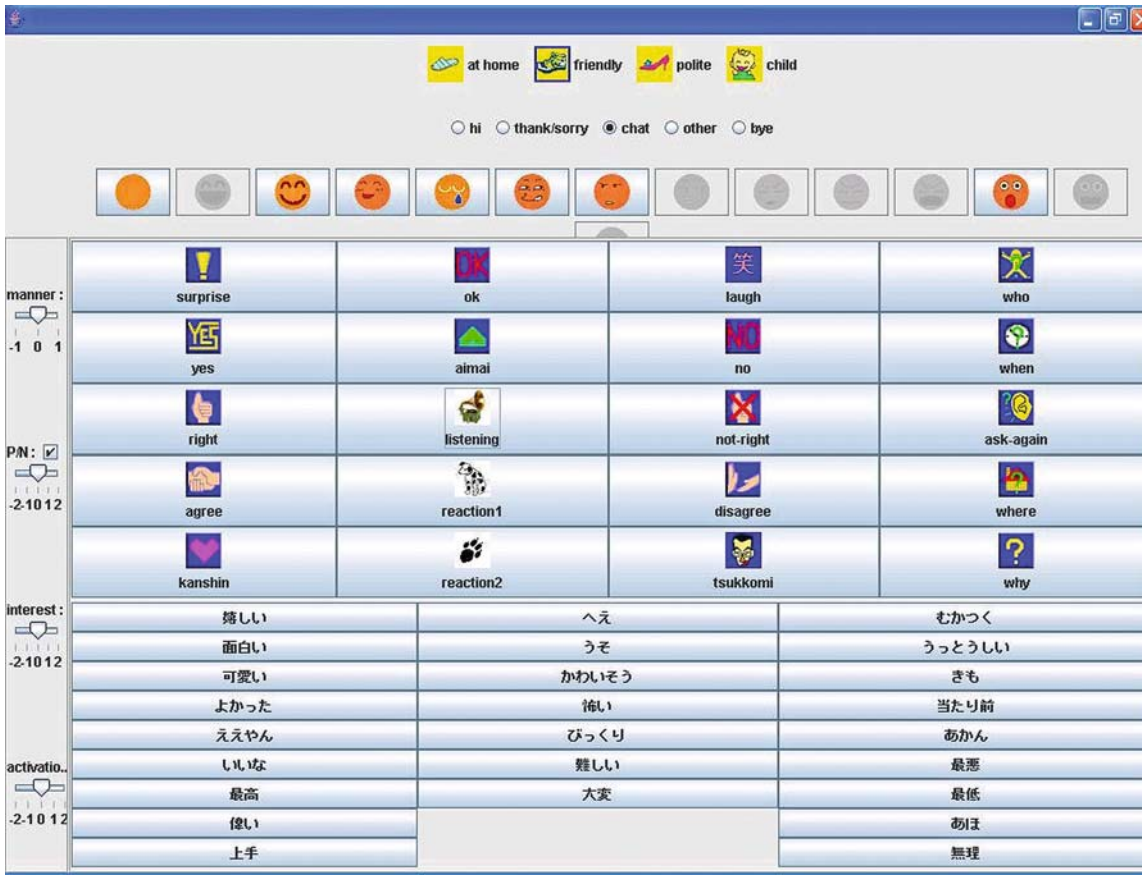


Fig. 25.4 The Chakai conversational speech synthesis interface. By clicking on a speech-act icon, a choice of emoticons is displayed in the *upper section* of the screen according to corpus availability, from which an utterance having the appropriate speech characteristics can be selected. Utterances are selected at random from among those in the same category within the corpus so that subsequent selection of the same combination will provide natural variety without unnecessary repetition

However, another paradox has arisen in its place. We originally believed that we would be able to capture truly natural and spontaneous emotional speech data by having a microphone active and in place before and while the emotional event took place. Instead, we find that by far the majority of our speech material is not marked for emotion as we then conceived it, but that it varies significantly in dimensions better related to affect and attitude, signaling the mood and interest of the speaker, his or her current relations with the listener, and controlling the variable flow of the discourse.

We started out by believing that emotion was the essential component lacking in our speech corpora for technology development, but we now consider that the human dimension that we were looking for is not best described by the term 'emotion' at all. Our data score very highly on the measure of paralinguistic to linguistic content described in the introduction, and are very far from the formal speech of less interactive situations, almost half being nonverbal and affect-related, but they lead us to conclude that the emotional state(s) of the speaker are not always directly expressed, and that social and interpersonal considerations override the supposed link between subjective emotion and displayed affective states. The social aspects of communication therefore take precedence over the blunt expression of feeling, and while the latter can perhaps be determined from an expressive utterance, the multiple levels of information in the former provide a richer source of data to be pro-

cessed if we are to better understand the person through her speech.

25.5.3 Concatenative Synthesis Using Expressive Speech Samples

With such a large corpus, including more than 600 h from one speaker alone, we were able to test some original approaches to concatenative synthesis of expressive speech. In this case, the units to be concatenated are no longer subsegmental, but can be as large as whole phrases. It might be debatable whether such a concatenative technique is still to be called speech synthesis, but since our unit-selection criteria make use of the labels described above, we would claim that this is still the case.

In parallel with the problem of determining the optimal unit size, is the equivalent problem of how to specify such units for input to the synthesizer. Plain text is no longer appropriate when the intention of the speaker is more important than the lexical sequence of the utterance. Instead, we needed to enable the user to quickly access a given corpus segment by means of higher-level intention-related constraints.

Figure 25.4 shows a recent proposal for Chakai, which is such a speech synthesis interface [25.40, 41] that allows for free input (by typing text into the white box shown at the bottom center) as well as the fast selection of various frequently used phrases and, in addition, an icon-based speech-act selection facility for the most common types of affective *grunt*. The name, not unrelated to CHATR, is composed of two Japanese syllables, meaning tea meeting, an event during which social and undirected chat is common. This format enables linking to a conventional CHATR-type synthesizer for creation of I-type utterances not found in the corpus, while providing a fast, three-click, interface for common A-type utterances which occur most frequently in ordinary conversational speech. Samples (and source code) are available from http://feast.atr.jp/chakai.

The selection of whole phrases from a large conversation speech corpus requires specification not just of the intention of the phrase (greeting, agreement, interest, question, etc.,) but also of the speaker's affective state (as desired to be represented) and the speaker's long- and short-term relationships with the listener at that particular time.

25.6 Conclusion

Humans are primarily social animals; they relate in groups and form close communities and subgroups. Much of human speech is concerned not with the transmission of propositional content or novel information, but with the transfer of affective information, establishing bonds, forming agreements, and reassuring each other of a positive and supporting (or other) environment.

In listening to a spoken utterance, human listeners parse not just its linguistic content, but also the way it has been spoken, voice qualities (including tone of voice) provide clues to its intent, in a way that is complementary to its content, to assist in the interpretation of the utterance.

In speech conversation, both the speaker and the listener are active at all times; the speaker predominantly so, conveying novel information, while the listener conveys feedback information about the flow of the discourse, and about the mutual states of comprehension.

In conversational speech, some of the time is devoted to imparting propositional content, but much of the time is devoted to managing the discourse; eliciting feedback, controlling turn-taking, and expressing affective states and stances.

In the synthesis of expressive speech, we must be able to generate sounds that reproduce all levels of information in the speech signal. At present there are no systems that can do this, but this is an active area of research and one for which there is a strong human need. If we can solve this problem, we will be able to produce a technology that people can relate to, can feel at ease with, and that can process information at levels far deeper than the mere linguistic.

References

25.1 G. Bailly, C. Benoit, T.R. Sawallis (Eds.): *Talking Machines: Theories, Models, and Designs, Reports Papers from the first ISCA Speech Synthesis workshop in Autrans* (North-Holland, Amsterdam 1992)

25.2 N. Campbell: Getting to the heart of the matter; speech as expression of affect rather than just text or language, Language Res. Eval. **39**(1), 109–118 (2005)

25.3 SSML, The W3 Speech Synthesis Markup Language: www.w3.org/TR/speech-synthesis/ (See also the papers from the 2005 SSML Meeting at http://www.w3.org/2005/08/SSML/Papers/)

25.4 N. Campbell, D. Erickson: What do people hear? A study of the perception of non-verbal affective information in conversational speech, J. Phonetic Soc. Jpn. **7**(4), 9–28 (2004)

25.5 I.G. Mattingly: Experimental methods for speech synthesis by rules, IEEE Trans. AU **16**, 198–202 (1968)

25.6 J. Allen: Linguistic-based algorithms offer practical text-to-speech systems, Speech Technol. **1**(1), 12–16 (1981)

25.7 K. Church: Stress assignment in letter to sound rules for speech synthesis. In: *ACL Proc. 23rd Annual Meeting*, ed. by University of Chicago (Association for Computational Linguistics, Chicago 1985) pp. 246–253

25.8 G. Akers, M. Lennig: Intonation in text-to-speech synthesis: Evaluation of algorithms, J. Acoust. Soc. Am. **77**, 2157–2165 (1985)

25.9 N. Campbell: Recording techniques for capturing natural everyday speech. In: *Proc Language Resources and Evaluation Conference LREC-02)* (Las Palmas, Spain 2002) pp. 2029–2032

25.10 N. Campbell: Speech and expression; the value of a longitudinal corpus. In: *Proc. Language Resources and Evaluation Conference* (2004) pp. 183–186

25.11 R. Cowie, E. Douglas-Cowie, C. Cox: Beyond emotion archetypes; Databases for emotion modelling using neural networks, Neural Networks **18**, 371–388 (2005)

25.12 K. Ishimura, N. Campbell: Telephone dialogue data base of JST/CREST expressive speech processing project, Proc. Ann. Conf. JSAI **16**, 147–148 (2002)

25.13 D. McNeill, F. Quek, K.-E. McCullough, S. Duncan, N. Furuyama, R. Bryll, X.-F. Ma, R. Ansari: Catchments, prosody, and discourse, Gesture **1**, 9–33 (2001)

25.14 R. Carlson, B. Granstrom: A text-to-speech system based entirely on rules, Proc. IEEE-ICASSP **76**, 686–688 (1976)

25.15 J. Allen, M.S. Hunnicutt, D.H. Klatt: *From Text to Speech, The MITalk System* (Cambridge Univ. Press, Cambridge 1987)

25.16 K. Hirose, K. Sato, Y. Asano, N. Minematsu: Synthesis of F0 contours using generation process model parameters predicted from unlabeled corpora: Application to emotional speech synthesis, Speech Commun. **46**(3-4), 385–404 (2005-2007)

25.17 A. Sakurai, K. Hirose, N. Minematsu: Data-driven generation of F0 contours using a superpositional model, Speech Commun. **40**(4), 535–549 (2003)

25.18 K. Silverman, M. Beckman, J. Pitrelli, M. Ostendorf, C. Wightman, P. Price, J. Pierrehumbert, J. Hirschberg: ToBI: A standard for labelling English prosody, Proc. ICSLP 92 **2**, 867–870 (1992)

25.19 The official ToBI website: http://www.ling.ohio-state.edu/ tobi/

25.20 Webpage of the Signal Processing Laboratory of the University of the Basque Country (UPV/EHU) in Bilbao: http://bips.bi.ehu.es/aholab/TTS/Expressive-Speech-Synthesis.html

25.21 P. Ekman: Basic emotions. In: *Handbook of Cognition and Emotion*, ed. by T. Dalgleish, M. Power (Wiley, New York 1999) pp. 301–320

25.22 K. Sjolander, J. Gustafson: Voice creation for conversational fairy-tale characters. In: *Proc. SSW SYnthesis Workshop* (2005)

25.23 A. Silva, G. Raimundo, C. de Melo, A. Paiva: To tell or not to tell... Building an interactive virtual storyteller. In: *Proc. AISB Symp. Language Speech and Gesture for Expressive Characters* (2004)

25.24 M. Theune, K. Meijs, D. Heylen, R. Ordelman: Generating expressive speech for storytelling applications, IEEE Trans. Audio Speech Language Process. **14**(4), 1137–1144 (2006)

25.25 N. Campbell: Recording techniques for capturing natural everyday speech. In: *Proc. Language Resources and Evaluation Conference* (Las Palmas, Spain 2002) pp. 2029–2032

25.26 J.E. Cahn: *Generating expression in synthesized speech*, M.S. Thesis (Massachusetts Institute of Technology, Cambridge 1989), http://alumni.media.mit.edu/ cahn/emot-speech.html

25.27 J.E. Cahn: From sad to glad: Emotional computer voices. In: *Proc. Speech Tech '88 Voice Input/Output Applications Conference and Exhibition* (New York 1988) pp. 35–37

25.28 J.E. Cahn: The generation of affect in synthesized speech, J. Am. Voice I/O Soc. **8**, 1–19 (1990)

25.29 J.E. Cahn: Generation of affect in synthesized speech. In: *Proc. 1989 Conf. American Voice I/O Society* (Newport Beach, California 1989) pp. 251–256

25.30 M. Bulut, S.S. Narayanan, A.K. Syrdal: Expressive speech synthesis using a concatenative synthesizer, Proc. ICSLP **2002**, 1265–1268 (2002)

25.31 MIT's Kismet (the expressiveness and richness of the robot's vocal modality and how it supports social interaction): http://www.ai.mit.edu/projects/sociable/expressive-speech.html

25.32 MIT Kismet and Affective Intent in Speech: http://www.ai.mit.edu/projects/sociable/affective-intent.html

25.33 A. Iida, N. Campbell, M. Yasumura: Design and evaluation of synthesised speech with emotion, J. Inform. Process. Soc. Jpn. **40** (1998)

25.34 A. Iida, N. Higuchi, N. Campbell, M. Yasumura: Corpus-based speech synthesis system

with emotion, Speech Commun. **40**(1–2), 161–187 (2002)
25.35 E. Eide, A. Aaron, R. Bakis, W. Hamza, M.A. Picheny, J.F. Pitrelli: A corpus-based approach to AHEM expressive speech synthesis. In: *Proc. 5th ISCA Speech Synthesis Workshop* (Pittsburgh, USA 2004)
25.36 J.F. Pitrelli, R. Bakis, E.M. Eide, R. Fernandes, W. Hamza, M.A. Picheny: The IBM expressive text-to-speech synthesis system for American English, IEEE Trans. Audio Speech Language Process. **14**(4), 1099–1108 (2006)
25.37 SVOX: http://www.svox.com/Innovation.aspx
25.38 The HUMAINE Portal – Research on Emotions and Human-Machine Interaction: http://emotion-research.net/
25.39 The Expressive Speech Processing project web pages: http://feast.atr.jp/
25.40 N. Campbell: Specifying affect and emotion for expressive speech synthesis. In: *Computational Linguistics and Intelligent Text Processing*, ed. by A. Gelbukh (Springer, Berlin, Heidelberg 2004)
25.41 N. Campbell: Conversational speech synthesis and the need for some laughter, IEEE Trans. Audio Speech Language Process. **14**(4), 1171–1179 (2006)

22. Linguistic Processing for Speech Synthesis

R. Sproat

An important part of any text-to-speech synthesis system is the linguistic processing component that takes input text and converts it into a feature representation from which actual synthesis can proceed. Linguistic analysis is hard, in a large measure because written language massively underspecifies linguistic information. This chapter reviews several issues in linguistic analysis starting from low-level text normalization issues, and ending with higher-level problems such as accent prediction and document-level analysis. We end with some prognosis of the future prospects for improvements over current technology.

22.1 Why Linguistic Processing is Hard

If one compares the problems faced by a text-to-speech (TTS) system to those faced by an automatic speech recognizer (ASR) system, one's first impression may well be that, compared with the situation with ASR, the language analysis portion of TTS has it rather easy. In an ASR system one has to reconstruct what was said from rather noisy input, so the language modeling problem can be formidable. For TTS, in contrast, one already knows what the words are since they, along with other linguistic information, are represented in the text. This view is a misconception.

While written language represents speech insofar as a literate speaker of a language can generally read a text aloud, writing is still a highly imperfect and incomplete representation. Writing is not transcription. Rather, writing presumes that the reader knows the language in question, and can fill in the details when the information provided by the written form is incomplete or misleading.

There are many such instances of incomplete or misleading information, ranging from abbreviations for which one must figure out the full word form, through homographs where one must choose among a set of differently pronounced words that happen to be spelled the same way, to the fact that no writing system has any systematic representation of phrasal prosody. In what follows we shall discuss each of these issues in turn, show why they are hard, and present some approaches that have been taken to solving them. We start with a brief reminder of how writing encodes linguistic information.

22.2 Fundamentals: Writing Systems and the Graphical Representation of Language

The basic input to a text-to-speech system is text, typically text that is encoded in the standard orthography of the language to be synthesized. Thus, the starting point of any discussion of linguistic processing in text-to-

speech synthesis should be an overview of how language is encoded in writing – or in other words how writing systems work. In this section we give a brief synopsis of writing systems, focusing on the issues that are of importance to the TTS problem.

As far as we know, writing was developed independently by exactly four civilizations: by the Egyptians around 3000 BC; by the Sumerians at around the same time; by the Chinese around 1500 BC; and by the Mayans around 1000 AD. Of these, only Chinese writing survives in an unbroken tradition into modern times in that a highly modified version of the original Shang dynasty script is used to encode modern descendants of the Chinese language spoken 3500 years ago. All writing systems encode pronunciation, but they do so to different degrees, and a number of writing systems also encode other, mostly semantic information. All ancient scripts – Mayan, Egyptian, Sumerian, and Chinese – encoded both sound and meaning; in modern Chinese, meaning is loosely encoded for most characters by means of *semantic radicals*, and sound is very roughly encoded in *phonetic components*. Among writing systems that putatively only encode sound, such as those based on the Greek-derived alphabets, there is a wide difference in the transparency of the encoding; for languages that use the Latin alphabet, Finnish is among the most straightforward in terms of letter–sound correspondence, and English is almost certainly the least straightforward.

However, the important point to bear in mind is that, while all writing systems encode sound and possibly other linguistic information, they are under no obligation to do so in a way that makes the recovery of that information easy. Writing is designed for people who know the language and can fill in the details. While there are certainly psycholinguistic consequences of different designs (there is a vast literature on this topic), as a practical matter it is relatively unimportant that Japanese *kanji* (Chinese characters) routinely have upwards of five distinct pronunciations; or that Arabic and Hebrew fail to represent most short vowels; or that distinctive stress placement information is not marked in Russian orthography. Each of these bits of missing information can generally be provided by someone who knows the language. Of course for a TTS system this is a challenge since it requires the system to model the language that native speakers all know.

TTS systems typically take their input in the form of electronic text, which of course obviates the need to deal with optical character recognition. But despite much progress on standardization by the unicode consortium, there are still areas where coding variants are allowed, leading to complications. For example, many Indian scripts have complex vowel symbols consisting of two or more separate components. The unicode standard permits more than one way of specifying these components, so that a system that deals with text in these languages must know about the legal variants. Furthermore, there are other variants which, though *illegal* are nevertheless found; Buckwalter [22.1] gives a nice overview of the complexities of Arabic text processing that are caused by the many variant encodings.

22.3 Problems to be Solved and Methods to Solve Them

22.3.1 Text Preprocessing

In Sect. 22.2 we introduced some of the ways in which writing encodes language but we tacitly assumed that we were dealing with ordinary words of the language spelled in the ordinary way.

But consider the following perfectly ordinary-looking paragraph taken from a story in the *New York Times*, April 11, 2006:

> *What made the* ***$12.08*** *transaction remarkable was that the customer was not just outside* ***Ms.*** *Vargas's workplace here on California's central coast. She was at a McDonald's in Honolulu. And within a two-minute span* ***Ms.*** *Vargas had also taken orders from drive-through windows in Gulfport,* ***Miss.****, and Gillette,* ***Wyo.***

In addition to the ordinary words and names such as one might expect to find in a dictionary (see Sect. 22.3.2), there are five items, marked in boldface, which are not ordinary words. The first is a numerical expression, more specifically a currency expression, which in this instance is to be read as *twelve dollars and eight cent*. The next four are abbreviations, namely two instances of *Ms.* for *Miz* (though note that this word has no conventional full-form spelling), one of 'Miss.' for *Mississippi* and one of *Wyo.* for *Wyoming*. Note that in this latter instance the period that marks the expression as an abbreviation also serves the double duty as the end-of-sentence marker.

A *text preprocessor* is the component of a TTS system that deals with analysis of *nonstandard words* such as the examples above, along with other low-level text-processing issues such as sentence segmentation and, in some languages, word segmentation. In this section we will review these issues, starting with the problem of end-of-sentence detection and word segmentation, broadly construable under the rubric of *tokenization*.

Tokenization

The first thing one needs to do in any text-preprocessing system is to divide the text into smaller units or *tokens* for further processing. In principle a token can correspond to any unit of linguistic structure, such as a word, a sentence or a paragraph. Most discussions of tokenization presume that the tokens are words, but this choice is somewhat arbitrary. We will therefore assume the broader definition and discuss the problem of dividing a text into sentences, and thence into words. These levels of tokenization will typically be further subdivided – words into morphemes, or sentences into phrases – and these issues will be discussed later in the chapter.

For sentence tokenization an ideal situation would be if there were a delimiter or set of delimiters that are used to delimit sentences and nothing else. A situation close to that obtains in Chinese, where declarative sentences are delimited by a small circle '。' that is used only for that purpose. In many languages, including English, the situation is less straightforward since the most common sentence delimiter, the period, has other functions including, most problematically, marking abbreviations; it can even serve both functions at once, as in the example of *Wyo.* above. Proper sentence tokenization is therefore intimately tied up with other aspects of preprocessing, such as abbreviation expansion. TTS preprocessors will therefore typically invoke abbreviation expansion before or concomitantly with sentence tokenization.

Tokenization into words is either relatively easy or relatively hard, depending upon which language one is dealing with. In English and many other languages, words are typically delimited by spaces or punctuation symbols and, while things are not always completely trivial, these orthographic conventions make the problem of word segmentation in such languages fairly easy. At the other end of the scale are languages like Chinese, Japanese, and Thai, where words are never delimited by spaces, and tokenization is therefore in principle fairly hard; in such cases punctuation usually *does* delimit words, but punctuation is of course relatively sparse and therefore does not provide much of a guide.

It is worth noting that the use or nonuse of spaces is mostly not *language* dependent, but *script* dependent. Languages that use alphabets such as Greek, Latin, Cyrillic will always mark word boundaries. This is also true of most scripts used for Indian languages. On the other hand, languages that use the Chinese writing system or derivatives including Chinese, Japanese, and traditionally Korean and Vietnamese do not use spaces. Only in the 20th century, with the wider use of Hangul did Korean introduce spacing to separate words (more accurately, accentual phrases or *eojeol*). On the other hand, even after abandoning the *chữ'-nôm* Chinese-derived writing system in favor of romanization in the 19th century, Vietnamese continues to this day not to mark word boundaries with spaces: spaces are used in Vietnamese not to mark word boundaries, but syllable boundaries [22.2]. A similar point can be made about the use of *capitalization*. Again, capitalization is not language dependent but script dependent. The Greek, Latin, Cyrillic, and Armenian scripts support capitalization, and languages using those scripts use capitalization to mark certain classes of words, typically names, words in sentence-initial position, words in titles, and so forth. Other scripts do not support capitalization.

Word segmentation has been studied extensively in Chinese and Japanese, less extensively in Thai, and even less in Vietnamese. Although there are important linguistic differences between these languages that make details of the problem different, for reasons of space we will restrict discussion here to Chinese.

If one can assume that one has a dictionary that lists all the words that one will find in a given text, the problem of segmentation is relatively straightforward. In *most* cases, simple algorithms like *maximum matching* work reasonably well: starting with a pointer at the left (or right) edge of the sentence, find the longest match in the dictionary, set the pointer after (before) that match, and iterate this process until you reach the end (beginning) of the sentence. However, such simple algorithms run afoul of cases where there is ambiguity. To handle such cases, people have typically used statistical language-modeling techniques, taking into account the probabilities of the words involved, and the transition probabilities between the words; class-based language modeling techniques, such as part-of-speech tag language models have also been used. It is important to realize though that, even with these techniques there are still cases that cannot be disambiguated. Such cases typically depend upon discourse or even real-world knowledge. *Gan* [22.3] discusses such cases. For example the sentence 马路上生病了 *mǎ lù shàng shēng bìng*

le can either mean "the horse became sick on the road" if the segmentation is *[mǎ] [lù] [shàng] [shēng bìng] le*, or "(someone) became sick on the road" if the segmentation is *[mǎ lù] [shàng] [shēng bìng] le*. Since both of these are valid interpretations and there is no way without further information to tell which one is more appropriate, it follows that ordinary language-modeling techniques cannot handle such cases.

One might wonder why, for Chinese TTS applications, one even cares about word segmentation. The reasons are that knowing what word one is dealing with and knowing where word boundaries fall are just as important in Chinese as they are in English. Many characters in Chinese (and orders of magnitude more in Japanese) have multiple pronunciations and in many cases the pronunciation depends upon what word the character occurs in. For example, in Mandarin the character 行 can be pronounced either as *xíng* as in 行人 *xíngrén* 'passer-by', or as *háng* as in 银行 *yínháng* 'bank'. Character pronunciations are also dependent upon word class. In traditional Chinese characters, as used in Taiwan, the character 乾 is pronounced as *gān* if it means *dry*, but *qián* if it means *heaven*. The former reading would be appropriate for the nonce compound 羊肉乾 *yángròu gān* 'mutton jerky', whereas the former would be appropriate in a personal name such as 林魁乾 *lín kuìqián*. This implies that for unknown words one needs in general to know more than just the sequence of characters is a word, but also what kind of word it is.

Another obvious motivation for word segmentation is prosodic phrasing (Sect. 22.3.3): it is usually desirable to avoid putting a prosodic break in the middle of a word.

Abbreviation Expansion

Abbreviation expansion might seem to be a rather easy problem. Faced with an abbreviation such as *kg.*, an English speaker will automatically think of *kilogram*. It is just a matter of table lookup. A moment's reflection, however, will reveal that things are not quite this simple. There are several issues that make the problem complicated.

- First, one must decide that one has an abbreviation as opposed to an ordinary word. Abbreviations are not always marked with a period, and even when they are, the period may be doing double duty as an end-of-sentence marker as in the example above. Deciding that *No* should be read as *North* in *No Arlington*, requires contextual information. Even if you are sure you have an abbreviation, there is the issue of how to read it. In [22.4] we carefully distinguished between three classes of abbreviation:
  - EXPN: terms that are to be *expanded* into full words, like *kg*
  - LSEQ: terms that are read as letter sequences, like *CIA*
  - ASWD: terms that are read as if they were an ordinary word, like *NATO*
- Any given abbreviation may be ambiguous, a standard example in English being *Dr.*, which can be *Doctor* or *Drive*.
- Even if you are sure which word is intended, you often need to predict the form of the word. Thus 1 kg is probably *one* ***kilogram***, but 2 kg is probably *two* ***kilograms***, whereas 2 kg *watermelon* is again *two* ***kilogram***. In a highly inflected language like Russian the situation is even worse. Thus the Cyrillic equivalent of *kg* can be expanded as *kilogram*, *kilograma*, *kilogramy*, *kilogramov*, *kilogramami* . . . , depending upon the context.
- People frequently make up new abbreviations, so you cannot count on being able to find all abbreviations in the dictionary. Novel abbreviations are especially rich in domains where there is a strong incentive to abbreviate, as in printed classified advertisements (where you are paying by the inch) or in the notes taken by customer service representatives (who have very little time to write down a summary of their conversations with customers). For example, in a database of AT&T customer care call summaries, we found no fewer than 18 distinct ways of abbreviating the word *customer*, only a handful of which could reasonably be considered standard.

Most systems use some version of table lookup. Sense disambiguation techniques such as the ones we discuss in Sect. 22.3.4 can be used for disambiguation.

Abbreviation expansion is an interesting and difficult problem and it is unfortunate that it has not received more systematic attention in the literature. The only such systematic study that we are aware of is work that we did during the 1999 Johns Hopkins Workshop on Speech and Language processing, and reported in [22.4]. In that paper, we used language modeling coupled with a channel model that predicted possible abbreviation expansions. An explicit goal was to deal with nonce abbreviations, such as the example of *customer* cited above. The most interesting examples in that work came from the domain of classified advertisements, where heavy abbreviation is typical, but there were sufficient examples of the full forms of words. For example, in addition to *brk*

apts one also found *brick apartments*; with a reasonable model of possible abbreviations, one could infer from the existence of *brick apartments* that *brk apts* might be a reasonable abbreviation of that, and so propose *brick apartments* as a possible expansion. An example of automatic abbreviation from a different domain, namely AT&T customer service transcriptions is shown below:

```
xplnd chrgs .. cust stated he w/
pay 26.45 & then w/ cancel his
srvc w/ att

explained charges .. customer
stated he will pay 26.45 & then
will cancel his service with att
```

Here full words such as *explained* and *charges* occur elsewhere in the corpus and are reasonable expansions for the abbreviated words *xplnd* and *chrgs*. Note that *w/* is used here both in its normal usage of *with*, as well as *will*, and that the expander gets these both correct.

Number Expression Expansion

Unlike other aspects of preprocessing, number expression expansion is fairly algorithmic – once you know which of the particular *uses* of the number expression is intended. To illustrate this point consider the following examples:

1. He saw **123** goats.
2. I live at **123** King Avenue.
3. I have never used Lotus **123**.

In case 1 the expression is being used to represent a normal number name *one hundred (and) twenty three*. In case 2, it represents a numerical *label*; in English such labels would typically be parsed into small chunks, so that a sequence *1234* would often be read as *twelve twenty-four*; in this case the sequence would be read as *one twenty-three*. In case 3, another kind of label, read as a sequence of digits – *one two three* – is intended. Deciding which of these is appropriate is a *sense disambiguation* problem, and techniques such as those discussed in Sect. 22.3.4 work reasonably well.

However, once one knows which type of number expression is intended, expansion is straightforward. For labels such as cases 2 and 3, one needs to parse the sequence into small chunks that are read as number names (as in case 1). The typical method for expanding number names in TTS systems is to write special-purpose rules, or to hard-code the expansion in a programming language such as C++. However, there is a theoretically more satisfactory method, namely to view the problem as a problem of *regular transduction*, implementable using weighted finite-state transducers (WFSTs); see Sect. 22.4 and [22.5]. Briefly, one can factor the problem into two components: the transduction of a digit sequence into a sum-of-products of powers of ten; and another transduction from such a factorization and the number name itself. The relatively few languages with productive vigesimal systems can also be handled straightforwardly by such methods.

What we have just described works reasonably well for languages such as English or Chinese where numbers are largely invariant in form. Languages such as Spanish where numbers inflect for grammatical categories such as gender add an additional layer of complexity: the WFST-based approach still works, but one must perform additional contextual analysis. A particularly complex case can be found in Russian, where number names inflect for gender and case. The form of nouns modified by number names is affected by both the case assigned to the noun phrase containing the number-name/noun combination, and by the value of the number itself; for details see [22.5].

22.3.2 Morphological Analysis and Word Pronunciation

Morphological analyzers have long been used in TTS systems as an aid to word pronunciation. One of the earliest morphological analyzers was DECOMP, part of the MITalk English TTS system [22.6]. DECOMP consisted of a finite-state model of morpheme combination, augmented with spelling change rules. DECOMP relied heavily on generative phonology in its implementation of morphology-dependent phonological alternations.

Word pronunciation frequently depends upon understanding the morphological composition of a word. If one blindly applies a general letter-to-sound transduction to the word *hothead*, one will likely pronounce the <th> sequence as /θ/; clearly one needs to know that the word is actually a compound out of two words, and that <th> should not be interpreted as a digraph.

In Russian, vowel pronunciation is heavily dependent upon stress placement in the word, and stress placement is in turn dependent upon morphological information. In the nominal declension system, for instance, placement of stress for a given form depends upon the particular declension class to which the noun in question belongs. Such information is not generally predictable from the phonological shape of the noun and seems to be an arbitrary property of the noun in question.

Morphology is not only relevant to the pronunciation of ordinary words, but also proper names; many per-

sonal names, for instance, have morphological structure that can be exploited in predicting pronunciation [22.7]. Thus, for instance, the pronunciation of a personal name beginning in *Mc* can often be predicted by separating off the *Mc*, looking up or otherwise predicting the pronunciation of the remainder, and reattaching the pronunciation of *Mc*. Morphological analysis of sorts is also needed for the correct pronunciation of many brand names, particularly in cases where capitalization or punctuation gives no hints as to decomposition. Recognizing that *espeech* in a URL is actually *e+speech* allows one to avoid a pronunciation such as /ɛspič/. Part-of-speech tagging (see Sect. 22.3.3) is another motivation for morphological analysis. Knowing that a form like *splurged* is probably a past tense or past participial verb form – and less likely to be a simplex noun – is useful information if one wants to assign part-of-speech information to a sentence containing this word.

Computational morphologists often distinguish between *lemmatization* (or *stemming*) and full-blown morphological analysis. A lemmatizer typically returns a pair consisting of a base (dictionary) form and a piece of morphosyntactic information: for example, *sat* might return (*sit*, PAST). Full-blown morphological analysis typically goes further and decomposes the word into smaller chunks, termed *morphs* or *morphemes*. A word such as *derivations* might be analyzed as *derive+ation+s*, with possible further annotations indicating the grammatical information about the pieces.

Most methods for full-blown decomposition have involved *finite-state transducers*, the most famous example of such a system being *Koskenniemi*'s [22.8] system for Finnish. Traditionally such systems have been handbuilt, a task that can obviously be very laborious for a language with a complex morphology. Recently, however, there has been a small amount of progress towards automating such acquisition in the form of a number of systems that can induce morphological alternations from unannotated corpora [22.9–11, inter alia]. However, such systems are still a long way from replacing linguistic experts in the construction of high-coverage morphological analyzers.

Lemmatizers have tended to use a variety of different methods, though all of these methods could, not surprisingly, be implemented using finite-state transducers. Some of the most interesting approaches to lemmatization include self-organizing methods such as those of *Yarowsky* and *Wicentowski* [22.12]. Part of this system is a decision tree that learns string suffixes of a word that are strongly associated with a particular suffix and stem change.

22.3.3 Syntactic Analysis, Accenting, and Phrasing

We have seen in various places in the preceding sections that written language is an imperfect representation of spoken language at the word level. Once one gets beyond the word level, this becomes even more true, especially when one considers the problems of *accenting* and *phrasing*. We discuss each of these topics in turn.

Accenting

In any language, different parts of a sentence are associated with different levels of *prominence*. So, for example, for the classic sentence *She had your dark suit in greasy wash water all year*, a fairly neutral reading of the sentence would have relatively high prominence on *had*, *dark*, *suit*, *greasy*, *wash* and *year*, and less prominence on the others. Such prominences are termed *accents*, and determining correct accent placement is a major issue in producing plausible-sounding synthetic speech. Accents themselves are typically implemented with some sort of *excursion* in fundamental frequency (F_0), possibly combined with an increase in amplitude and other phonetic effects.

The linguistic literature on accent often divides languages into three different basic classes; see [22.13–15, inter alia]. *Stress languages*, of which English is an example, have prominences associated with words, but how these are expressed in terms of pitch excursions varies with the context. In *pitch accent languages*, of which Japanese is an example, words are lexically marked for both accent placement and the F_0 excursion shape. Finally, in *tone languages* most or all syllables are lexically marked for F_0 information. Nevertheless, *all* languages will assign different prominence to words under different conditions, though the details of how this works out vary a lot from language to language [22.14, 15].

In the work on accent assignment in English, people have often made a three-way distinction between words that are fully accented, words that are deaccented, and words that are *cliticized*. Typically, deaccented words are not associated with an F_0 excursion. Cliticized words, in addition, tend to be prosodically very weak, with the consequence that they are quite short in duration, and behave as if they were really a part of an adjacent word.

A simple model of accentuation would make the decision on accenting on the basis of part-of-speech information. Typically, nouns, verbs adjectives, and other *open-class* words tend to be accented. Func-

tion words – determiners, pronouns, articles, etc. – tend to be deaccented or, if they are short, cliticized. A minimal requirement for accent assignment, then, is a part-of-speech tagger, or at least a word classifier that can distinguish between open- and closed-class words. Part-of-speech (POS) tagging is a well-studied problem [22.16–19, inter alia], and modern POS taggers perform quite well, with error rates in the range of perhaps 3% for newswire-type text, and any number of methods can be used in the context of TTS.

Of course, POS-based prediction can only get you so far: the purpose of accenting, at least to a large extent, is not to communicate broad lexical categories, but rather to communicate the importance of information. Indeed, it is a consequence of that fact that major lexical categories such as nouns tend to be accented, and minor categories such as prepositions tend to be deaccented: nouns typically carry more information than and are therefore more important than prepositions. So, much of the work on accenting has focused on ways to predict the information content. Early work such as [22.20] considered various categories of terms, including cue phrases (*And then* ...), and items that are *new* in the discourse, which tend to be given special prominence – versus those that are *given* and tend to be given less prominence. More recently, corpus-derived information-theoretic measures have been used to predict accent [22.21].

In English, a particularly interesting and complex set of cases involve complex noun phrases, i. e., a noun preceded by one or more adjectival or nominal modifiers. In a discourse-neutral context, some instances are accented on the final word (*Madison* ***Avenue***), some on the penultimate (***Wall*** *Street*, *kitchen* ***towel*** *rack*), and some on an even earlier word (***sump*** *pump factory*). For a binary nominal the primary determinants of accenting are largely semantic, although there is a fair amount of purely lexical specification [22.22–24]. So, right-hand accent is often found in cases where the left-hand element denotes a substance out of which the second element is made (cf. *apple* ***pie***, *steel* ***bar***), but there are numerous often quite systematic (***banana*** *bread*) exceptions. Computational models, e.g., [22.25, 26], have been somewhat successful at modeling the semantic and lexical generalizations; for example [22.26] uses a combination of hand-built lexical and semantic rules, as well as a statistical model based on a corpus of nominals hand-tagged with accenting information. Accenting on nominals longer than two words is generally predictable given that one can compute the nominal's structure (itself a nontrivial problem), and given that one knows the accentuation pattern of the binary nominals embedded in the larger construction [22.23, 26, 27].

Phrasing

Speakers normally break up long sentences into a sequence of several prosodic phrases. There are two basic varieties of prosodic phrase commonly recognized in the literature: an utterance is typically assumed to consist of one or more *intonational phrases*, with each of these in turn consisting of one or more *intermediate phrases*. A linguistic analyzer for a TTS system must compute appropriate places to insert phrase boundaries.

If punctuation is used liberally so that there are relatively few words between the commas, semicolons or periods, then a reasonable guess at an appropriate phrasing would be simply to break the sentence at the punctuation marks, although this is not always appropriate. The real problem comes when long stretches occur without punctuation; in such cases, human readers would normally break the string of words into phrases, and the problem then arises of where to place these breaks. The simplest approach is to have a list of words, typically function words, that are likely indicators of good places to break [22.28, 29]. One has to use some caution however, since while a particular function word like *and* may coincide with a plausible phrase break in some cases, in other cases it might coincide with a particularly poor place to break, e.g., *I was forced to sit through a dog* ***and*** *pony show that lasted most of Wednesday afternoon.*

Most prosodic phrasing prediction systems attempt to do some amount of syntactic analysis of the input, ranging from full-blown parsing, to partial parsing to simply computing phrasing on the basis of part-of-speech tag sequences. An early system that used the output of a syntactic parser in phrasing prediction was the system reported in [22.30]. *Bachenko* and *Fitzpatrick* employed a wide-coverage deterministic syntactic parser (FIDDITCH, [22.31]) to construct a syntactic analysis for a sentence; the syntactic phrases were then transduced into prosodic phrases using a set of heuristics, including heuristics based on observations, originally due to *Gee* and *Grosjean* [22.32], that phrases tend to be somewhat balanced in length.

However, most systems have tended to eschew full-blown parsing, partly because of the computational overhead of parsing, and partly because the benefits of having a full parse have not been adequately

demonstrated. [22.33] reported on a corpus-based statistical approach that used classification and regression trees [22.34] to train a decision tree on transcribed speech data. In training, the dependent variable was the human prosodic phrase-boundary decision, and the independent variables were generally properties that were computable automatically from the text, including part of speech sequence around the boundary, shallow-parsing information such as the location of the edges of long noun phrases, the distance of the boundary from the edges of the sentence, and so forth.

More recently *Taylor* and *Black* [22.35] used generative Markov models to predict the most likely sequence of phrase breaks given a part of speech sequences. The Markov models also modeled phrase balancing heuristics of the kinds used in [22.30]. They experimented with various Markov model topologies, tag sets and smoothing methods, and concluded that the best setup allows for a 79% correct identification of breaks in a held out corpus, which is competitive with other approaches in the literature.

22.3.4 Sense Disambiguation: Dealing with Ambiguity in Written Language

Automatic sense disambiguation has become an important area of research in natural language processing, with a number of *bake-offs* (SENSEVAL and SEMEVAL), and the development of a number of sense-tagged corpora, and resources such as WordNets in many languages [22.36, 37].

Sense disambiguation has applications to several problems in TTS, but the most obvious application is to *homograph* disambiguation. For example, it is generally useful to know whether the word *mole* should be pronounced /mol/ (as in the mammal, chemistry or spy sense) or as /mole/ (as in the Mexican sauce sense); similarly, one wants to know if *IV* should be read as *four* (or *fourth*), or as *I. V.* (as in *IV drip*). In Japanese, one wants to know if the character *mountain* should be read as *san* or *yama*. In Russian one wants to know if *goroda* should be read with stress on the first syllable (*of a city*) or on the last (*cities*).

Sense disambiguation is essentially a language modeling problem. The writer of a text presumably knows which sense of each word is meant, so what they are trying to communicate can be thought of as coming with sense tags. What they actually produce, however, is written in an orthographic representation where distinct senses of words are conflated into a single orthographic form. It is the task of the reader, or the TTS algorithm, to reconstruct the underlying sense. As with language modeling more generally, a number of approaches have been taken, but particularly popular approaches involve *discriminative* methods that aim at finding contextual features that are particularly relevant to discriminating the two senses. The first such approach to be applied in TTS was due to *Yarowsky* [22.38, 39], who applied a more-general sense-disambiguation method based on *decision lists*. Features of the context, such as words within a particular window of the target word, words in a particular position relative to the target word, the part-of-speech sequence surrounding the target word, are examined individually. Features are ranked by the *log likelihood ratio* of the two senses, with the highest log likelihood ratios being assigned to those features that are the best discriminators. The list of features rank-ordered by log likelihood ratio is then used as a decision list, where the decision for a particular target word is decided on the basis of the first matching feature. A fragment of a decision list for *bass*, from [22.38] is given in Table 22.1. Since Yarowsky's work there has been a lot of research on sense disambiguation including a special issue of the journal *Computational Linguistics* [22.36], and several workshops, such as [22.37], and a number of different approaches have been explored. While most of this work has not specifically addressed the question of homograph disambiguation for TTS, any of these techniques could be so applied.

Table 22.1 Decision list for *bass*, (after Yarowsky)

Decision list for *bass* (*highly abbreviated*)		
log *L*	**Evidence**	**Class**
10.98	*fish* in ±*k* words	⇒ bæs
10.92	*striped* bass	⇒ bæs
9.70	*guitar* in ±*k* words	⇒ beIs
9.20	bass *player*	⇒ beIs
9.10	*piano* in ±*k* words	⇒ beIs
9.01	*tenor* in ±*k* words	⇒ beIs
8.87	*sea* bass	⇒ bæs
8.49	*play/V* + bass	⇒ beIs
8.31	*river* in ±*k* words	⇒ bæs
8.28	*violin* in ±*k* words	⇒ beIs
8.21	*salmon* in ±*k* words	⇒ bæs
7.71	*on* bass	⇒ beIs
5.32	bass *are*	⇒ bæs

22.4 Architectures for Multilingual Linguistic Processing

There are naturally many ways in which one could implement any of the methods that we have been discussing, ranging from implementing special-purpose toolkits, to hard-coding many of the details of the algorithms in a programming language such as C++ or, in the open-source Festival system scheme [22.40].

There is some advantage to having a single computational mechanism for handling most or all aspects of text processing in all languages. One approach that has gained in popularity over the past decade are methods based on weighted finite-state transducers (WFSTs) [22.5, 41, 42]. Work on multilingual TTS was pioneered in the Bell Labs, but the approach was also adopted by the commercial system under development at Rhetorical, as well as in research systems such as the DeKo system at Stuttgart (http://www.ims.uni-stuttgart.de/projekte/DeKo/). The appeal of WFSTs is that they offer a common framework for handling a wide variety of phenomena that can be viewed as string-to-string transduction problems. Through the regular operations of union, concatenation, Kleene closure, and composition, complex systems can easily be constructed out of simple pieces. The algebraic and algorithmic properties of WFSTs are well understood [22.43–45]. Furthermore, many toolkits exist for compiling WFSTs out of grammars such as might be written by a linguistic expert – such as the Xerox/XRCE Toolkit [22.46, 47], van Noord's toolkit, or Lextools [22.5, 48]. WFSTs can also be constructed via a variety of statistical language-modeling methods. Since the regular operations listed above apply in the same way independently of how the WFSTs were constructed, one can have a single text-analysis module that covers all languages: one merely needs to plug in a new set of WFSTs constructed for the language of interest.

There are some *disadvantages* to using WFSTs as opposed to, for example, a system that is carefully hard-coded in a language such as C++. WFST models tend to be larger and may also be slower than such hard-coded systems, and this has led some system developers to eschew the use of general-purpose solutions. But given the move to ever-larger unit-selection databases for the production of the speech output, and the huge increases in computational resources that have become available, this view seems shortsighted. Indeed, WFSTs have been used in a number of commercial systems, including the systems once sold by Lucent Speech Solutions and Rhetorical Systems. The fact that these systems are no longer sold has nothing to do with technical reasons, and everything to do with the vagaries, and in the former case incompetence, of the business world.

22.5 Document-Level Processing

We have focused in this article on linguistic processing, but properties of the document are also important in conveying information effectively using speech. Different regions of a document often need to be rendered in different ways, and computing these regions is thus important for appropriate rendering.

For example, suppose one is reading an email message. In plain-text portions of the message one can ignore line feeds since they convey no information. But this is not the case in other portions of the message, for example in a signature block as in Fig. 22.1. Indeed, not only are line feeds significant – e.g., one would normally want some sort of break between the name *Charles Davies* and the following line – layout is also important. It would be wrong, for example, to read the line "Bell Laboratories, Lucent Technologies | tel (908) 582-1234" as one unit, since the portion to the right of the '|' is actually part of its own block, which includes the telephone, fax, and email address. As with text preprocessing, there has been relatively little sys-

```
Charles Davies
Dialogue Modeling Research Department
Multimedia Communications Research Laboratory
Bell Laboratories, Lucent Technologies  |tel (908) 582-1234
600 Mountain Avenue, Room 2d-500        |fax (908) 582-4321
Murray Hill, NJ 07974, USA              |ced@bell-labs.com
http://www.bell-labs.com/noname/mcs/
```

Fig. 22.1 A sample signature block

tematic research on document-level processing, but one strand of research was work done at Bell Labs and reported in [22.49, 50]. This work combined statistical approaches to detecting different types of text regions (plain text, tables, signature blocks, etc.) with two-dimensional layout analysis for the complex structures in signature blocks.

A special note should be made here on the influential work of *T. V. Raman* [22.51] on the audio rendering of different classes of text, in particular mathematical expressions. Raman's work is particularly focused on auditory rendering for blind users, but of course has much broader applicability to any situation where the user cannot easily work with a visual display.

22.6 Future Prospects

Linguistic analysis is a critical component of a text-to-speech system. While much of the attention in TTS in recent years has been focused on synthesis techniques that improve voice quality, voice quality is not enough on its own for a system to be both natural sounding and pleasing to listen to. Supposing that one has a completely natural-sounding synthesizer in terms of voice quality, but that the system routinely expands abbreviations incorrectly, frequently chooses the wrong sense of a homograph, mispronounces words, and places inappropriate emphasis and phrase breaks; the best one could say about such a system is that it sounds like someone with a very nice voice, but whose brain is disconnected with reality.

Fortunately this hypothetical situation does not occur in general: over the past few decades, substantial improvements have been made in all aspects of linguistic analysis for TTS, and some of this progress has been reviewed here. However, even though TTS systems today can deal in a sensible way with a wide variety of both unstructured and structured text, there is still much that such systems cannot handle. Some examples of problems that as yet have no general solution are as follows:

- *Discourse-level reasoning*: For an example like *¿Cuantas tortas quiere Ud.? 200.* ('How many cakes do you want? 200.'), Spanish speakers would have no problem producing the correct form of *200 – doscientas*, the feminine form, since *tortas* (*cakes*) is feminine. In order to get this right, one must deduce the coreference relation between *200* and *tortas*. There has of course been a substantial amount of work on reference resolution (see [22.52] for a recent collection of papers), but the problem is hard, and we are far from a solution that covers all cases.
- *Emotion*: When humans read text, they frequently convey affect. This is especially true in genres, such as fiction, where it would generally be quite inappropriate to read the text in a dull-sounding voice that might be more fitting for a list of financial figures. There has been a substantial amount of work on the acoustic correlates of emotion (Chap. 25), but there has been a lot less on how one might predict appropriate emotion from text. One recent strand of work [22.52–54], has used machine-learning methods to attempt to predict the appropriate emotion given features computable from the input text. Still, it will likely be a long while before we will have systems that can perform at all like humans on this problem.
- *Accent and intonation*: Despite the massive amount of work on accenting and intonation, it is still very hard to predict appropriate accent placement, accent type, and intonational parameters from the text. Some of this relates to emotion, which we have just discussed, but not all of the issues are so readily characterized. Classic cases such as the following, which humans would typically have no problem with given appropriate context, are beyond the capabilities of the linguistic analysis components of current TTS systems.

1. *John called Bill a Republican, and then he insulted him.* (Interpretation: after calling Bill a Republican, John insulted Bill.)
2. *John called Bill a Republican, and then **he** insulted **him**.* (Interpretation: John called Bill a Republican, which is interpreted as an insult, so Bill insulted John back.)

By default TTS systems are likely to produce the first accentual variant. In a similar vein current TTS systems are incapable of deciding on the basis of textual analysis that the word *what* in the following example should reasonably be pronounced with an incredulous sounding intonation:

- *Marge ate the whole jar of habaneros.*
- *She did **what**?*

Ultimately the problem comes down to the point that we noted in the introduction: in order to read a text, one needs a very good model of the underlying language, and this is precisely what TTS systems lack.

To drive home this point with a final example, consider the following piece of dialog spoken by the character Jim in *Huckleberry Finn*:

> *"Doan' hurt me – don't! I hain't ever done no harm to a ghos'. I alwuz liked dead people, en done all I could for 'em. You go en git in de river agin, whah you b'longs, en doan' do nuffn to Ole Jim, 'at 'uz awluz yo' fren'." (Mark Twain, Huckleberry Finn, Chapter 8).*

Twain's spelling is intended to represent what he calls the Missouri Negro dialect. What is interesting about this example, and others like it, is that you do not actually have to know the dialect that Twain was trying to represent in order to be able to figure out what Jim is saying. Any competent speaker of English could read this text and understand it, and could at least read the words out in their own pronunciation, and if they have at least some familiarity with the dialect region, might even be able to use Twain's orthographic cues to make a guess as to how Jim might have said them. Of course, you are probably thinking that it would be straightforward to build a model that, given appropriate training data, could learn the relevant mappings between Jim's speech and standard English. But this is beside the point: competent speakers can generally adapt quite quickly to such nonstandard spellings without any training whatsoever in the particular spelling system being used. A robust knowledge of the underlying language is enough to overcome noisy input of this kind. Such ambient noise rarely throws off human speech recognition, though it can completely incapacitate ASR.

Just as with ASR, which has arguably reached the limits of what one can do with the current technology, so with linguistic processing for TTS my sense is that we will only see incremental improvements in the short term with tweaks to the current approaches. Underexplored problems will, of course, continue to show improvement as more researchers become interested in them. Emotion prediction from text, for example, is one area where we might expect to see substantial progress over the next few years. Richer datasets with annotations for discourse-level phenomena may, if developed, allow for some improvements in these areas too. Generally, however, we will fail to significantly close the gap between machine and human ability. Entirely new approaches to language understanding will be needed before that can occur. Needless to say, if and when such new approaches are developed they will have applicability well beyond TTS.

References

22.1 T. Buckwalter: Issues in Arabic morphological analysis. In: *Arabic Computational Morphology: Knowledge-Based and Empirical Methods*, ed. by A. Soudi, G. Neumann, A. van den Bosch (ACM, New York 2006)

22.2 W. Hannas: *Asia's Orthographic Dilemma* (University Hawaii Press, Honolulu 1997)

22.3 K. W. Gan: Integrating word boundary identification with sentence understanding, Ph.D. Dissertation (National University of Singapore, Singapore 1995)

22.4 R. Sproat, A. Black, S. Chen, S. Kumar, M. Ostendorf, C. Richards: Normalization of non-standard words, Comput. Speech Lang. **15**(3), 287–333 (2001)

22.5 R. Sproat (Ed.): *Multilingual Text-to-Speech Synthesis: The Bell Labs Approach* (Kluwer, Boston 1997)

22.6 J. Allen, M.S. Hunnicutt, D. Klatt: *From Text to Speech: The MITalk System* (Cambridge Univ. Press, Cambridge 1987)

22.7 C. Coker, K. Church, M. Liberman: Morphology and rhyming: Two powerful alternatives to letter-to-sound rules for speech synthesis, Proc. ESCA Workshop on Speech Synthesis, ed. by G. Bailly, C. Benoit (ESCA, Autrans 1990) pp. 83–86

22.8 K. Koskenniemi: Two-level morphology: a general computational model for word-form recognition and production, Ph.D. dissertation (University of Helsinki, Helsinki 1983)

22.9 J. Goldsmith: Unsupervised acquisition of the morphology of a natural language, Comput. Linguist. **27**(2), 153–198 (2001)

22.10 P. Schone, D. Jurafsky: Knowledge-free induction of morphology using latent semantic analysis, Proc. Comput. Nat. Lang. Learning Conf., Lisbon (2000) pp. 67–72

22.11 D. Yarowsky, R. Wicentowski: Minimally supervised morphological analysis by multimodal alignment, Proc. ACL-2000, Hong Kong (2001) pp. 207–216

22.12 K. Vijay-Shanker, C.-N. Huang: Minimally supervised morphological analysis by multimodal alignment, Proc. 38th Meeting of the Association for Computational Linguistics, Hong Kong (2000) pp. 207–216

22.13 J. Pierrehumbert, M. Beckman: *Japanese Tone Structure*, Linguistic Inquiry Monograph Series (MIT Press, Cambridge 1988)
22.14 R. Ladd: *Intonational Phonology* (Cambridge Univ. Press, Cambridge 1996)
22.15 C. Shih: *Prosody Learning and Generation* (Springer, Berlin, Heidleberg 2007)
22.16 K. Church: A stochastic parts program and noun phrase parser for unrestricted text, Proc. Second Conf. Applied Natural Language Processing (ACL, Austin 1988) pp. 136–143
22.17 D. Cutting, J. Kupiec, J. Pedersen, P. Sibun: A practical part-of-speech tagger, Proc. Third Conf. Applied Natural Language Processing (1992)
22.18 E. Brill: A simple rule-based part of speech tagger, Proc. Third Conf. Applied Natural Language Processing (ACL, Trento 1992)
22.19 A. Ratnaparkhi: A maximum entropy part-of-speech tagger, Proc. First Empirical Methods in Natural Language Processing Conference, Philadelphia (1996)
22.20 J. Hirschberg: Pitch accent in context: Predicting intonational prominence from text, Artificial Intelligence **63**, 305–340 (1993)
22.21 S. Pan, K. McKeown: Word informativeness and automatic pitch accent modeling, EMNLP/VLC 99 (Association for Computational Linguistics, College Park 1999)
22.22 E. Fudge: *English Word-Stress* (Allen Unwin, London 1984)
22.23 M. Liberman, R. Sproat: The stress and structure of modified noun phrases in English. In: *Lexical Matters*, ed. by A. Szabolcsi, I. Sag (CSLI University of Chicago Press, Chicago 1992)
22.24 G. Cinque: A null theory of phrase and compound stress, Linguistic Inquiry **24**(2), 239–297 (1993)
22.25 A. Monaghan: Rhythm and stress-shift in speech synthesis, Comput. Speech Lang. **4**, 71–78 (1990)
22.26 R. Sproat: English noun-phrase accent prediction for text-to-speech, Comput. Speech Lang. **8**, 79–94 (1994)
22.27 M. Liberman, A. Prince: On stress and linguistic rhythm, Linguistic Inquiry **8**, 249–336 (1977)
22.28 D. Klatt: Review of text-to-speech conversion for English, J. Acoust. Soc. Am. **82**, 737–793 (1987)
22.29 D. O'Shaughnessy: Parsing with a small dictionary for applications such as text to speech, Comput. Linguist. **15**, 97–108 (1989)
22.30 J. Bachenko, E. Fitzpatrick: A computational grammar of discourse-neutral prosodic phrasing in English, Comput. Linguist. **16**, 155–170 (1990)
22.31 D. Hindle: A parser for text corpora. In: *Computational Approaches to the Lexicon*, ed. by B.T.S. Atkins, A. Zampolli (Oxford Univ. Press, New York 1994)
22.32 J.P. Gee, F. Grosjean: Performance structures: A psycholinguistic and linguistic appraisal, Cognitive Psychology **15**, 411–458 (1983)
22.33 M. Wang, J. Hirschberg: Automatic classification of intonational phrase boundaries, Comput. Speech Lang. **6**, 175–196 (1992)
22.34 L. Breiman, J. Friedman, R. Olshen, C. Stone: *Classification and Regression Trees* (Wadsworth Brooks, Pacific Grove 1984)
22.35 P. Taylor, A. Black: Assigning phrase breaks from part-of-speech sequences, Comput. Speech Lang. **12**, 99–117 (1998)
22.36 N. Ide, J. Véronis: Word sense disambiguation: The state of the art (1998) pp. 1–4
22.37 P. Edmonds, R. Mihalcea, P. Saint-Dizier (Eds.): *ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions* (Association for Computational Linguistics, Philadelphia 2002)
22.38 D. Yarowsky: Three machine learning algorithms for lexical ambiguity resolution, Ph.D. dissertation (University of Pennsylvania 1996)
22.39 D. Yarowsky: Homograph disambiguation in text-to-speech synthesis. In: *Progress in Speech Synthesis*, ed. by J. van Santen, R. Sproat, J. Olive, J. Hirschberg (Springer, New York 1997) pp. 157–172
22.40 P. Taylor, A. Black, R. Caley: The architecture of the Festival speech synthesis system, Proc. Third ESCA Workshop on Speech Synthesis, Jenolan Caves (1998) pp. 147–151
22.41 R. Sproat: Computational morphology. In: *Handbook of Natural Language Processing*, ed. by R. Dale, H. Moisl, H. Somers (Dekker, New York 1997), forthcoming
22.42 W. Skut: Finite-state text processing in a speech synthesis system. In: *Language Technology for Business Applications*, ed. by J. Piskorski, A. Przepiorkowski (Poznan 2004)
22.43 M. Mohri: Finite-state transducers in language and speech processing, Comput. Linguist. **23**, 1 (1997)
22.44 F. Pereira, M. Riley: Speech recognition by composition of weighted finite automata. In: *Finite-State Language Processing*, ed. by E. Roche, Y. Schabes (MIT Press, Cambridge 1997)
22.45 M. Mohri, F. Pereira, M. Riley: A rational design for a weighted finite-state transducer library, Lecture Notes in Computer Science, Vol. 1436 (1998)
22.46 R. Kaplan, M. Kay: Regular models of phonological rule systems, Comput. Linguist. **20**, 331–378 (1994)
22.47 K. Beesley, L. Karttunen: *Finite State Morphology*, CSLI Publications (University of Chicago Press, Chicago 2003)
22.48 M. Mohri, R. Sproat: An efficient compiler for weighted rewrite rules, 34th Annual Meeting of the Association for Computational Linguistics (ACL, Santa Cruz 1996) pp. 231–238
22.49 R. Sproat, J. Hu, H. Chen: EMU: An e-mail preprocessor for text-to-speech, IEEE Signal Processing Society 1998 Workshop on Multimedia Signal Processing, Los Angeles (1998)

22.50 H. Chen, J. Hu, R. Sproat: E-mail signature block analysis, ICPR'98, Brisbane (1998)
22.51 T.V. Raman: Audio system for technical readings, Ph.D. dissertation (Cornell University 1994)
22.52 S. Harabagiu, D. Farwell (Eds.): *Workshop on Reference Resolution and its Applications, ACL 2004, Barcelona* (2004)
22.53 C. O. Alm, D. Roth, R. Sproat: Emotions from text: machine learning for text-based emotion prediction, HLT/EMNLP 2005, Vancouver (2005)
22.54 C. O. Alm, R. Sproat: Emotional sequencing and development in fairy tales, First Int. Conf. Affective Computing and Intelligent Interaction, Beijing (2005)

23. Prosodic Processing

J. van Santen, T. Mishra, E. Klabbers

Speech synthesis systems have to generate natural-sounding speech output from text. One of the key aspects of speech is prosody, which must be both natural (i. e., sounding like a human) and meaningful (i. e., sounding like a human who understands the contents of the text). The computation of prosody from text can be divided into the computation of prosodic tags from text and the computation of acoustic speech features from these tags. This chapter focuses on the latter. It provides an overview of prosody in human–human communication, including the communicative functions of prosody and the acoustic correlates. Discussed next is a historical overview of the various methods that have been used for prosody generation in speech synthesis, as well as of current methods. Special attention is paid to prosody generation in unit selection synthesis methods, in which large corpora are searched for fragments of speech that match the phonemes and prosodic tags computed from text and that optimize various cost functions, and in which prosody is not modeled and speech not modified. We conclude the chapter by advocating hybrid approaches in which search capabilities of unit selection methods are combined with the speech modification methods from more-traditional approaches.

23.1 Overview

In natural speech, speakers use a multitude of acoustic cues to communicate a message. Some of these cues signal the identities of speech sounds, or phonemes. Additionally, however, there are cues that signal stressed syllables, prominent words, phrase boundaries, sentence mode, metalinguistic features such as irony, and the emotional state of the speaker. For synthetic speech to be fully comprehensible and sound natural, all these functions of prosody must be mimicked. Attention must be paid to how to compute *prosodic tags* such as syllable stress from text, to which *acoustic prosodic features* are used to express these tags (e.g., pitch, timing), and to the precise manifestation of these features (e.g., the shape of the pitch contour). Recent systems differ fundamentally from older systems in how they perform this mimicking task, and there is no reason to assume that future systems will not differ fundamentally from current systems; we will therefore look both into the past and into the future. This chapter will focus on the computation of acoustic manifestations from prosodic tags.

23.1.1 What Is Prosody?

Prosody is studied in a range of areas, including linguistics (both at the phonetic and the phonological level), speech pathology, and speech technology; in addition,

we can speak about prosody at the speech production, acoustic, and perceptual levels. As a result, terms do not always have the same meaning in different contexts. In our specific context, however, we can define prosody rather precisely, by explicitly referring to text-to-speech (TTS) components and their internal representations.

We will assume that a TTS system has a *text analysis component*, whose task it is to create a potentially very rich linguistic data structure, perhaps aided by markup of the input text, that contains tags (*target tags*) that characterize input text in terms of phoneme labels, parse trees, part-of-speech labels, syllable stress, and a range of additional features, perhaps even including emotional state or pragmatic features. Generally speaking, all tags except those referring to, or exclusively used for the computation of, phoneme identities can be called prosodic. Systems diverge in how these target tags are used to generate output speech. In some systems, such as unit selection systems, a tagged speech corpus is searched for units that match the target tags. In other systems, such as traditional diphone systems, a *target prosody component* computes from these target tags a specification of the output prosody, or *target prosody*, such as the duration of phonetic segments in ms or the pitch in Hz, which can be overlayed onto selected units by various means.

23.1.2 Prosody in Human–Human Communication

The role of prosody in human–human communication is manifold. First, acoustic prosodic features are used to distinguish between lexical items. Examples include pitch in tone languages, duration in languages that distinguish between long and short vowels or that have geminates, and both features – as well as additional features – to create different stress patterns for words that contain the same phonemes. Second, acoustic prosodic features are used to structure utterances in terms of phrases and to indicate relationships between phrases in utterances. For example, utterances containing lists differ in prosodic structure from utterances containing parenthetic remarks. Third, prosody is used to focus attention on certain words for a variety of purposes, such as highlighting a contrast (contrastive stress), emphasizing their importance, or enhancing their intelligibility because they are unpredictable from the context. All these uses share the property that they help the listener understand the contents of the message.

These roles are fairly straightforward, and can be mimicked reasonably well by current text analysis systems. Roles that have traditionally been ignored in TTS systems have to do with *pragmatic* and *affective* uses that go beyond helping to convey the literal contents of the message. There is a vast array of such uses in human–human communication. In fact, one should take note that, at the start of life, in the pre-verbal child, *all* communication is purely prosodic. It is thus not surprising that prosody continues its important role in communication throughout life. Examples include the communication of social roles and authority, making statements about the statement made (e.g., irony, doubt, sarcasm, firmness), instilling an emotion in the listener (e.g., reading a bedtime story to a child, or threatening someone by tone of voice), and, of course, expressing emotions. The reasons for this neglect may include both the difficulty of computing the relevant tags from text and the typical usage of TTS systems – for example, no sarcasm is desired for systems that provide flight arrival times.

We will now briefly review the literature on acoustic features of prosody that is of direct relevance for TTS.

Timing

Research of timing in human speech has typically focused on the durations of phonetic segments, less often of syllables or words, or sub-phonemic units. The reason for this is practical rather than fundamental. For example, boundaries between most segments are easy to determine, with the exception of transitions between segments having the same (e.g., n–m) or similar (e.g., vowel to glide) manners of production. At the end of this subsection we will revisit the issue of what the proper unit of timing is. Another limitation of this research is its neglect of pragmatic and affective prosody.

Most studies have focused on vowel duration in read speech. Two methodologies are commonly used. One is to design a balanced text corpus that allows one to test the presence of effects of a specific tag, e.g., word length (contrasting, e.g., brave, braver, bravery). The other method is to take natural text and use sophisticated statistical methods to predict durations from the tagged recorded speech. Both methodologies have shown that vowel duration depends on at least half a dozen factors, including: *identity-related tags* such as the identity of the vowel and of the surrounding (in particular following) consonants; *stress-related tags* such as vowel stress and word prominence; and *positional tags* such as the position of the syllable in the word and in the utterance structure. What is extremely important is that, together, these tags can predict vowel duration rather well. In read speech, a large number of studies have

shown vowel duration can be predicted with mean errors in the 10–20 ms or 15–25% range; correlations between observed and predicted durations tend to be in the 0.80–0.90 range [23.1–4]. Consonant durations, which require slightly different tags (such as within-syllable location: coda versus onset), have provided similar results [23.5].

In other words, durations of phonetic segments can be predicted accurately from text, using as criteria overall statistics such as the median absolute difference between observed (in a speech corpus) and predicted durations. There are at least two important caveats here, however. One is that such criteria may hide audible flaws in specific subsets of the data. For example, when we modify predictions by reducing the predicted duration of utterance-final vowels by 50%, there will be a pronounced effect perceptually but it will have little impact on the statistics because utterance-final vowel tokens are only a small percentage of the collection of all vowel tokens in a corpus. This suggests using a minimax error criterion (which minimizes the maximum error), after splitting up the data in appropriate subsets. Second, as shown by *Kato* et al. [23.6], listeners may be less attuned to absolute durations than to relative durations of neighboring units. This suggests using an error criterion that depends on specific durational ratios.

Why study durations of phonetic segments? One idea that has been discussed is to study the durations of longer units such as words or syllables. The merit of this idea for the spurposes of TTS is limited, because, even if it were the case that the durations of these longer units are extremely well behaved and hence highly predictable (e.g., exact isochrony at the syllable or foot level), one still needs to specify durations at the subunit level. For example, *van Santen* and *Shih* [23.7] showed that in stressed versus unstressed syllables mostly the onset and nucleus are stretched whereas in phrase-final versus phrase-medial syllables, primarily the nucleus and the coda, are stretched. In addition, one may even have to pay attention to timing at the sub-phonemic level, because it is also known that in contrasts such as mend versus meant mostly the final part of the nucleus as well as the nasal coda consonant are stretched, whereas in bide versus bite the early part of the diphthong is stretched [23.8].

Finally, the concept of a timing *unit*, whatever its size, inaccurately presupposes that the underlying articulatory gestures are always in lock-step with each other. The reason for neglecting this point of view in TTS is, of course, that articulatory timing does not easily map onto concatenative TTS architectures, or at least those architectures that compute target prosody and then impose the target timing on stored units by compressing or expanding them but not by modifying their spectral trajectories in ways that are reflective of articulatory asynchronies.

In summary, while in terms of overall statistical criteria the prediction of durations looks like a solved problem, several issues remain, including the search for perceptually more valid error criteria and a better understanding of sub-phonemic timing, in particular at the articulatory level.

Pitch

Research on pitch in human–human communication goes back far longer than research on timing because it can be studied – certainly qualitatively – without digital methods for speech segmentation. Because of this long history, but also because this area is fraught with controversy and different schools of thought, we will not attempt to review the vast literature on this topic here. Instead, we will review only some of the basic facts about pitch.

Pitch is the perceptual correlate of fundamental frequency F_0. Intonation refers to patterns of pitch (at the perceptual level) or F_0 (at the production level) in speech. F_0 is more difficult to measure than segment duration for a variety of reasons. First, phonatory state (e.g., creaking, breathiness, low harmonic-to-noise ratio) may cause pitch tracking errors. Second, F_0 is absent in unvoiced regions, including in parts of nominally voiced regions that are often devoiced (e.g., voiced fricatives). Third, there are important segmental perturbations even in regions where F_0 can be reliably extracted. For example, vowel–nasal or nasal–vowel transitions are often accompanied by short-lived pitch movements of up to 10 Hz. Also, during the first 50 ms of a post-obstruental vowel, F_0 may be increased by 25 Hz or more. Finally, high vowels are associated with higher pitch values than low vowels, by as much as 10 Hz.

These difficulties result in F_0 contours that are often inaccurate and, even if accurate, are hard to interpret in terms of which features reflect intended prosody rather than segmental articulatory effects. What makes matters particularly complex is that these segmental articulatory effects are not random noise that can be removed by smoothing but have locally systematic effects that – if one were to make the traditional assumption that pitch peaks are critical for prosodic interpretation – may produce spurious pitch peaks or obscure true pitch peaks.

In part to deal with these difficulties, a number of descriptive schemes have been proposed. Some of these describe F_0 contours in terms of a small num-

ber of *pitch targets* [23.9]. A labeler, who is acutely aware of these measurement and interpretation difficulties, can reliably label a contour in terms of such targets. A different approach is to stylize F_0 contours using a sequence of straight line segments. If done carefully, these line segments may indeed cut through the regions polluted by segmental articulatory effects and generate a more-interpretable processed contour. Finally, quantitative models have been proposed that fit the entire curve; by having a relatively small number of parameters, the fitted curve will not mimic the fine details of the observed curve, and thereby – typically, but not necessarily – eliminate the segmental effects.

Of course, these schemes are not merely methods for generating more-robust descriptors of F_0 contours, but also constitute intonational theories, ranging from purely nonprocess phonological linguistic theories [23.9] to theories about the physiology of phonation [23.10]. While discussing these theories in detail is beyond the scope of this chapter, it must be made clear that these theories have implications for how one processes speech in TTS. For example, systems based on tones and break indices (ToBI= [23.11] – an intonational labeling scheme based on Pierrehumbert's work – compute *abstract intonational tags* (such as H^* for a high tone associated with a stressed syllable) from text, and then either compute target prosody from these labels (in traditional diphone synthesis) or use the labels to search a speech corpus (in unit selection synthesis). This intermediate layer of abstract intonational tags is absent in typical applications of the Fujisaki model, where the values of the Fujisaki model's parameters are predicted directly from tags in the linguistic data structure relating to standard features such as syllable position, syllable stress, and other tags that do not include abstract intonational tags such as in ToBI.

We end this subsection by briefly listing a few of the basic intonational phenomena in standard US English. First, the global shape of F_0 contours. F_0 contours typically show local maxima close to stressed syllables. There is also usually a globally downward trend of the F_0 contour over the duration of a phrase. This downward trend is reversed in the final syllable or syllables in yes/no questions or in nonterminal phrases, but further accelerates downward in terminal phrases. In yes/no questions, there is a sharp rise at the end of the phrase; in nonterminal phrases, there is often a slight rise, or rise–fall–rise pattern (*continuation rise*) at the end. A further downward trend phenomenon occurs when speakers produce a list of items; here, typically each list item is presented in a register lower than that of the preceding item. A TTS system that mimics these basic shapes correctly can be considered adequate for conveying nonpragmatic or affective prosody. However, even a cursory inspection of affectively or pragmatically laden speech reveals an array of more-complex F_0 phenomena. For example, stressed syllables may be associated with local F_0 minima; there may exist F_0 plateaus, or regions with a global upward trend, that persist for several words. Generally, current TTS systems other than special unit selection based systems that have the right recordings (e.g., of emotional speech) do not mimic these phenomena.

A second basic phenomenon of F_0 is perceptual sensitivity to small changes in how pitch movement is aligned with the syllable or segmental boundaries [23.12, 13]. In studies of this phenomenon, it was found that changes of less than 100 ms in alignment were not only audible but also changed perceived meaning.

In summary, mimicking natural intonation is seriously challenged by the limited knowledge we have about natural intonation. Many challenges remain regarding how to measure and characterize pitch contours, how they vary as a function of prosodic content, and how intonation is perceived.

Other Acoustic Features

Most TTS systems are concerned with timing and intonation. However, there is strong evidence that other acoustic features play a role in prosody as well. One of these has to do with the overall spectral shape of speech, reflecting multiple aspects of speech production such as phonation and lower-jaw position. For example, at the end of an utterance subglottal pressure tends to decrease and the vocal chords – specifically changes in the open quotient – contribute to a more-breathy voice quality. Also, a lowered jaw position results in a large oral aperture, which in turn results in less attenuation of higher frequencies.

A second feature has to do with reduction phenomena. Reduced vowels tend to have formant trajectories that are more-heavily influenced – coarticulated – by neighboring phonemes, and may as a result deviate substantially from their target values. These deviations contribute to the perception of a reduced syllable as being unstressed [23.14]. Likewise, reduced consonants may have less-complete closure in combination with less subglottal pressure, which leads to frication of stops and to *muffled* fricatives.

It is not certain how important the contributions of these features are to overall perceived naturalness of syn-

thetic speech. However, it is important to keep in mind that, even if they play a background role in affectively and pragmatically neutral speech, their role in charged speech is likely to be considerable, in particular for features such as harshness of voice, tenseness, and other phonatory features.

23.2 Historical Overview

Before giving an overview of current approaches to prosodic processing, we briefly mention older approaches in order to set the stage for a discussion of the fundamental challenges that these current approaches are facing (Sect. 23.3).

23.2.1 Rule-Based Approaches in Formant Synthesis

The first computer-based speech synthesis systems used rules to generate speech. For duration modeling, the best-known type of rule-based approach is a sequential rule system as applied by *Klatt* [23.15, 16] in the MITalk system [23.17]. His model assumes that

1. each phonetic segment type has an inherent duration specified as one of its distinctive features,
2. each rule results in a percentage increase or decrease in the duration of the segment, but
3. the segment cannot be compressed shorter than a certain minimum duration.

The rules were derived from small speech production experiments and the literature. More-detailed information about the Klatt duration model can be found in Sect. 23.4.1.

For intonation modeling, a well-known rule-based approach is presented by *Pierrehumbert* [23.9, 18]. This approach describes the intonation contour as a series of target values. Because of the declination effect, the F_0 range narrows and drifts downwards over the course of the phrase. Targets are viewed as belonging to two categories, high and low. When two targets are both high and are sufficiently separated in time, the computer program computes a sagging contour between them. Otherwise, the F_0 contour between the two targets is a monotonic curve. More-detailed information about Pierrehumbert's approach can be found in Sect. 23.4.2.

23.2.2 Statistical Approaches in Diphone Synthesis

In the past two decades advances in computer technology have led to the use of statistical approaches to model prosody. These approaches use large databases of speech to train prosodic models. For duration modeling, a well-known model is the sums-of-products approach [23.1] as implemented in the Bell Labs multilingual speech synthesis system [23.19]. This approach takes advantage of the fact that most interactions are directionally invariant, which allows these interactions to be described with equations consisting of sums and products. Phonemes that are affected similarly by the various factors are grouped into subclasses. The decisions concerning this grouping are based on exploratory data analysis and phonetic/phonological literature. For each subclass of segments, a separate sums-of-products model is trained. Exploratory data analysis is an essential step in order to appropriately decide which factors are important and how many levels on a factor should be distinguished. For each subclass this can lead to different results. Thus, a great deal of phonetic and phonological knowledge can be incorporated in the model. More-detailed information about the sums-of-products model can be found in Sect. 23.4.1.

A well-known statistical approach for modeling intonation is the tilt model [23.20]. In the tilt model, intonation is characterized by a sequence of phonetic intonational events. Like in Pierrehumbert's model these events are pitch accents and boundary tones. Each event has a fall and rise component which can vary in size and can be described by five *tilt* parameters. Speech corpora are analyzed automatically to find events and corresponding tilt parameters. A classification and regression tree (CART) can then be used to retrieve tilt parameters for synthesis [23.21]. More-detailed information about the tilt model can be found in Sect. 23.4.2.

23.2.3 Using as-is Prosody in Unit Selection Synthesis

The next step in speech synthesis development was to use large corpora of natural speech not just for training prosody models, but also as a source of units for concatenative synthesis. Instead of having one token for each diphone, the corpus contains several tokens with different phonetic and prosodic context characteristics.

One of the first systems to use such a unit selection synthesis approach was the CHATR system [23.22, 23]. The phoneme durations and F_0 values at 10 ms intervals are used as target values when searching for the most-optimal units. The best units are then concatenated and no further prosodic modification takes place. More-detailed information about prosody in unit selection synthesis can be found in Sect. 23.4.2.

23.3 Fundamental Challenges

23.3.1 Challenge to Unit Selection: Combinatorics of Language

As has been pointed out by several authors [23.24, 25], TTS systems have to face the challenge of generating speech for text that has not been pre-recorded.

For unit selection based systems, the combinatorial challenge is far more severe, because, in the absence of prosodic modification of the stored speech, recordings are needed of each unit in a large number of prosodically distinct contexts. The analyses by *van Santen* [23.24] show that, for unrestricted domain synthesis, the chance that the units needed to render an arbitrary sentence are available in the desired prosodic contexts in even a very large corpus is vanishingly small.

This does not mean that there are no combinatorial challenges for target prosody-based systems that modify the prosody of stored speech. In fact, because the number of combinations of prosodic tags is extremely large, a premium is put on the *generalization capabilities* of the statistical methods used for, e.g., duration prediction. This also means that one can deceive oneself severely when a system is evaluated on a test text corpus that is not too different from the corpus used for statistical estimation, because the test text corpus may have substantial overlap of prosodic context *types*. It has been shown [23.2] that statistical approaches that impose constraints on durational behavior, such as quasi-linear regression based methods (e.g., sums-of-products models; [23.2]) fare far better than approaches that do not impose constraints, such as classification and regression trees [23.26]. These constraints can be of several types. For example, the constraint of *directional invariance* states that, e.g., for two occurrences of the same vowel in two contexts that only differ in stress, the stressed occurrence will be longer in duration. In other words (where $D()$ indicates duration, v a vowel, S stress, and $c_1, \cdots, c_n$ other prosodic tags),

$$D(v, c_1, \cdots, c_n, +S) \geq D(v, c_1, \cdots, c_n, -S) \,.$$

A different constraint states that the ordering of durations over all prosodic contexts is the same for each phoneme within a given class (e.g., voice fricatives). For example (where PF denotes phrase finality),

$$\begin{aligned} D(+S, +\mathrm{PF}) &\geq D(-S, +\mathrm{PF}) \\ &\geq D(+S, -\mathrm{PF}) \\ &\geq D(-S, -\mathrm{PF}) \,. \end{aligned}$$

These constraints have the effect of providing reasonable extrapolation or interpolation to new types.

23.3.2 Challenge to Target Prosody-Based Approaches: Multitude of Interrelated Acoustic Prosodic Features

While avoiding the combinatorial explosion inherent in the requirement for unit selection-based approaches to have recordings of all units in a great variety of prosodic contexts, target prosody-based approaches face three different and equally severe fundamental problems. First, the target prosody is synthetic, not natural. Thus, the generated pitch contours are not guaranteed to have natural shapes, and the rhythm may be off.

Second, this approach allows the TTS system to only mimic those features of acoustic prosody that are explicitly modeled. Thus, if only F_0 and duration are computed, then the output speech may have inappropriate voice quality or loudness.

Third, the signal processing needed to impose target prosody on recorded units is well known to often cause audible distortions.

23.4 A Survey of Current Approaches

With these inherent limitations in mind, we now turn to a survey of current approaches to prosodic processing.

23.4.1 Timing

In this section we will give an overview of the predominant approaches to duration modeling.

The Klatt Duration Model

The Klatt duration model was developed by Dennis Klatt at MIT in the 1970s and 1980s [23.27]. It is part of the MITalk formant synthesizer [23.17]. The Klatt model is a sequential rule-based model. The model assumes that

1. each phonetic segment type has an inherent duration that is specified as one of its distinctive properties,
2. each rule tries to effect a percentage increase or decrease in the duration of the segment, but
3. segments cannot be compressed shorter than a certain minimum duration [23.15].

The model is summarized by the following equation:

$$\mathrm{DUR} = \mathrm{MINDUR} + \frac{(\mathrm{INHDUR} - \mathrm{MINDUR})\mathrm{PERC}}{100}, \tag{23.1}$$

where MINDUR is the minimal duration of the segment in an *accented* condition, INHDUR is the inherent duration of a segment in milliseconds, and PERC is the percentage shortening or lengthening determined by applying a set of rules in sequence. Klatt used a set of 10 rules that relate to effects of the phonetic environment, emphasis, stress level, etc. on the current segment's duration. Each rule adjusts the PERC term multiplicatively and the final result is the product of the rules plus the effect of one final rule that is applied after the calculation of DUR [23.17].

The problem with this type of approach is that the parameter values are based on small-scale studies and findings reported in the literature in which the factors pertaining to a particular rule are varied while most others are kept constant. Since there may be interactions between contextual factors, the obtained parameter values are likely to be inaccurate for contexts where the factors held constant have different levels. Models of this type have been developed for several languages including American English [23.17, 28], Swedish [23.29], German [23.30], and French [23.31].

Classification and Regression Trees (CARTs)

CARTs were first introduced by *Breiman* et al. [23.32], and are tree-based nonlinear regression algorithms that have proven successful for a variety of prediction problems where there are nonlinear relations between the the variable you are trying to predict and the predictors. *Riley* [23.26] proposed the use of CART to model phoneme durations for text-to-speech synthesis. The CARTs can be constructed automatically. A binary-branching tree is grown with an algorithm that accepts phoneme labels with correct durations expressed as z-scores from a set of training data. The tree construction method looks for binary splits determined by a single factor that best correlates with the context by minimizing the variance of the estimated error within each cluster after the split. The result is a tree with questions at the node pertaining to the place and manner of articulation of current and surrounding segments, the position of a segment in the syllable, word, or phrase, or the lexical stress or accent on a syllable, etc., and with predicted duration values at the leaves. At synthesis time, the tree is traversed for each phoneme until a predicted z-score duration value is found. The main concern with training a CART is that the tree can become so large that it is overtrained, in which case it matches the training data well, but does not perform so well on new test data. Overtraining can be prevented by pruning the tree. A cross-validation scheme is used in which the tree is grown on 9/10 of the available data and tested on 1/10 of the data which is repeated 10 times for different subsets and averaging their variance. Branches that do not fit the unseen data well are then pruned. Because the procedure is completely automatic, CARTs are a very popular method of duration prediction. There are many programs available to train CARTs, including the *Wagon* procedure that is part of Festival [23.22]. However, overtraining or lack of coverage of the relevant phonetic/prosodic contexts may lead to suboptimal prediction. The CART approach to duration modeling has been used in TTS systems of several languages including Korean [23.33], Czech [23.34], Hindi and Telugu [23.35], and Italian [23.36]. The correlation between observed and predicted durations for these systems lies between 0.6 and 0.8 and the root-mean-squared error (RMSE) lies between 20 and 27 ms.

Neural Networks

One well-known example of a neural-network-based approach to duration modeling is the syllable-based approach developed by *Nick Campbell* as part of his

PhD thesis at the University of Edinburgh in the early 1990s [23.37]. The assumption is that neural networks can learn the underlying interactions between the contextual effects. The model predicts syllable durations first and then fits phoneme durations to the syllables. For each syllable a feature vector is computed which consists of information about the number of phonemes in the syllable, the nature of the syllabic peak (or nucleus, e.g., reduced, lax, tense vowel), the position in the tone group, the type of foot, stress level, and word class (function versus content word).

The syllable duration is calculated without prior knowledge of phonetic constraints. The phoneme durations within each predicted syllable class are then determined using an elasticity principle. For each phoneme class, there is a distribution of log durations that resembles a normal distribution and that is characterized by a mean duration μ and a standard deviation σ. In the majority of cases, the amount of compression or lengthening within a syllable can be expressed with a single constant k. This relies on the assumption that all segments in a syllable are lengthened or shortened equally in terms of standard deviation. Thus, the duration of the i-th segment is given by $\exp(\mu_i + k\sigma_i)$, where k is computed such that the following expression matches the precomputed overall syllable duration:

$$\sum_{i=1}^{n} \exp(\mu_i + k\sigma_i) \,. \tag{23.2}$$

This approach does not take the segmental make-up of the syllable into account, which can contribute considerably to the overall syllable duration and to the amount of lengthening applied to one particular phoneme.

The Sums-of-Products Approach

The sums-of-products approach was developed by *van Santen* et al. [23.2] as part of the Bell Labs multilingual text-to-speech synthesis system [23.19]. A sums-of-products model (23.3) gives the duration for a phoneme/context combination, as described by the feature vector $\boldsymbol{f}$. K is a set of indices, each corresponding to a product term. I_i is the set of indices of factors occurring in the i-th product term. In other words, a sums-of-products model is an equation whose parameters correspond to factor levels, which are combined by taking products of subgroups of parameters and then adding the products. Two examples of sums-of-products models are the additive model (where $K = \{1, \ldots, N\}$ and $I_i = \{i\}$) and the multiplicative model (where $K = \{1\}$ and $I_1 = \{1, \ldots, N\}$):

$$\mathrm{DUR}(\boldsymbol{f}) = \sum_{i \epsilon K} \prod_{j \epsilon I_i} S_{i,j}(f_j) \,. \tag{23.3}$$

To build a sums-of-products model for a TTS system, a large text corpus is used and all information relevant for duration prediction is computed and encoded in feature vectors. A set of sentences covering a subset of feature vectors is selected using a greedy algorithm. These sentences are then recorded and segmented into phonemes. Statistical analyses are carried out on the phoneme durations to manually construct a *category tree* in which all phonemes are grouped into subclasses that are affected similarly by contextual factors (not like in CART where phonemes with similar durations are grouped together) and to construct a set of sums-of-products models for each of the leaves in the tree. The statistical analyses ensure that the category tree and sums-of-products models best represent the data with as few parameters as possible.

The training is an iterative process in which exploratory data analysis is used to determine which levels have to be distinguished on a given factor for each phoneme category and which factors interact. By discarding irrelevant factors and factor levels, the number of missing data in the database will decrease and the number of observations per factor level will increase. By collapsing the correct factor levels, the predictions will be more accurate. The decisions are based on a number of information sources:

1. phonological/phonetic knowledge
2. the corrected means that are computed to determine the effect size of each factor level
3. the two-way corrected means that are computed to determine whether factors interact
4. the outcome of the multiplicative model in terms of prediction error

The calculations take place in the logarithmic domain to reduce the skewness of the duration distribution and to allow a more-accurate prediction of small durations.

The sums-of-products approach has been used for many different languages including American English, German ([23.3], $r = 0.90$, RMSE $= 19$ ms), Mandarin Chinese ([23.4], $r = 0.87$, RMSE $= 26$ ms), and Dutch ([23.38], $r = 0.77$, RMSE $= 23$ ms). *Maghbouleh* [23.39] has shown that the sums-of-products approach requires less training data than CART to achieve comparable results and performs better on unseen test data.

23.4.2 Intonation

There are many intonation models and theories, however, not all of them have been used for TTS. In this section, we will only present models that have been used in TTS synthesis systems. We will outline the implicit assumptions underlying each model, the explicit representation of F_0 chosen by each model, and the way in which each model can be used to compute intonation in a text-to-speech system. The presented models cover a wide spectrum of intonation approaches. We will be describing two phonological tone sequence models (the Pierrehumbert theory, the ToBI-based approaches), a phonetic sequential model that is not tone inspired [the recurrent neural network (RNN) model], a phonetic sequential model that involves acoustic stylization (the tilt model), one perceptually motivated acoustic stylization model [the Institute of Perception Research (IPO) model], three phonetic superpositional models [the Fujisaki model, the linear alignment model, the superposition of functional contours (SFC) model], one prosodic model (the Kiel model), and one physiological model [the soft template markup language (STEM-ML) model].

Pierrehumbert's Theory of Intonation

The Pierrehumbert theory of intonation [23.9], developed by Janet Pierrehumbert in 1980 as part of her doctoral dissertation, is a phonological model of intonation. It is based on autosegmental-metrical (AM) phonology [23.40, 41]. In keeping with the AM theory, the Pierrehumbert model considers intonation to be a sequence of high (H) and low (L) tones. The H and L are in *phonological opposition*, i. e., the difference in sound between them serves to distinguish intonational meaning. The two types of tones never interact with each other, rather they follow each other sequentially in an utterance.

The H and L tones are the building blocks of three larger tone units: *pitch accents*, *phrase accents*, and *boundary tones*. *Pitch accents* mark prominence. They are either single tones (H∗, L∗), or pairs of tones (L+H∗, L∗+H, H+L∗, H∗+L); the ∗ denotes the alignment of the tone with a stressed syllable. One or more pitch accents comprise an *intermediate phrase*. One or more intermediate phrases comprise an *intonational phrase*, the largest prosodic unit posited by this theory. The edges of the intonational phrase are marked by *boundary tones*. The boundary tones are single tones (%H, %L, H%, L%); the % denotes the alignment of the boundary tone with the pitch onset or offset of the intonational phrase. Pitch movement between a pitch accent and a boundary tone is indicated by a *phrase accent* (H−, L−), denoted by the diacritic, −.

To ensure that the model renders well-formed intonational representations, Pierrehumbert defined a finite state grammar that specifies the combinations in which pitch accents, phrase accents, and boundary tones can occur. She also devised a set of *phonetic realization rules* [23.18] to produce F_0 contours from the phonological model of intonation described above.

Generating the F_0 contour of a target utterance in a TTS system using the Pierrehumbert theory of intonation involves three main steps: first, determine the tonal representation of the utterance using the finite state grammar. Second, specify the target F_0 values of the high and low tones depending on the metrical prominence of the associated syllables, and the F_0 values of the preceding tones using the phonetic realization rules. Third, using the rules again, connect the target F_0 values to generate a F_0 contour: if two neighboring targets are far apart, connect them with a sagging contour implemented via a quadratic function. Otherwise, connect them via monotonic curves.

ToBI-Based Approaches

ToBI stands for *tones and break indices* [23.11]. Based on Pierrehumbert's theory of intonation, it was developed in four research meetings between 1991 and 1994 as a standard for describing American English intonation. It has since been extended to transcribe other languages and dialects [23.42–44].

ToBI consists of three parallel labeling tiers. The first tier is the *tone tier*. The tones specified by Pierrehumbert's theory are labeled in the tone tier. The second tier is the *break index tier*. In the break index tier, *break indices*, ranging from 0 to 4, are marked. *Break indices* mark the boundary strength between adjacent words; 0 indicates no boundary, 3 indicates an intermediate phrase boundary (- in Pierrehumbert's model), and 4 indicates a intonational phrase boundary (% in Pierrehumbert's model). The third tier is a miscellaneous tier, where hesitations, disfluencies, laughter, non-speech sounds, etc., are labeled.

It is important to note that ToBI is a labeling system and it *does not* specify the means to produce quantitative intonation from the ToBI labels. However, there are both rule-based and statistically trained approaches that can be applied to the ToBI labels to generate F_0 contours. An example of the rule-based approach is Jilka's handcrafted rule system for specifying the F_0 contour of American English from ToBI labels [23.45].

Jilka's approach is similar to Pierrehumbert's *phonetic realization rules*: rules specify the target F_0 values associated with ToBI labels, depending on pitch range and the voiced part of the syllable. The target F_0 values are calculated from left to right, taking into account only preceding ToBI labels, not subsequent ones. The F_0 contour is produced by linear interpolation between target points.

An example of the statistically trained approach is *Black* and *Hunt*'s linear-regression-based approach for generating F_0 contours from ToBI labels [23.46]. This approach simply involves predicting three target F_0 values for every syllable, one at the start of the syllable, one at mid-vowel position, and one at the end of the syllable, by means of linear regression. The prediction formula is defined as,

$$F_0 = I + w_1 f_1 + w_2 f_2 + w_3 f_3 + \ldots + w_n f_n \; . \tag{23.4}$$

The f_i variables indicate the features that contribute to the F_0 value of a syllable, such as ToBI label associated with the syllable, syllable position in the phrase, syllable stress etc., I and w_i are parameters that are estimated by linear regression.

The RNN Intonation Model

The recurrent neural network (RNN) intonation model, developed by *Traber* in 1991 [23.47, 48], uses *neural networks* to predict the F_0 contour. A *neural network* can be considered to be a nonlinear statistical model with many parameters. These parameters are estimated in the training phase so that they can return an optimal set of outputs from the corresponding input. The neural-network-based approach was motivated by the goal of using minimal human effort to obtain high-quality intonation, i. e., while humans would specify *which* phonological units were relevant for the phonetic realization of intonation, clever machine learning techniques would figure out *how* the phonological units map to the F_0 contour.

In the implementation of this intonation model in the SVOX TTS system [23.49], the F_0 contour was generated per syllable. For each syllable, the accent value of the syllable and segmental properties relating to its position in the phrase and the sentence, as well as the accent values of the neighboring syllables are given as input to the neural network. The network outputs the F_0 region related to the syllable (represented by 8 samples of the F_0 contour). The resultant F_0 regions are concatenated together to produce the complete F_0 contour.

The Tilt Intonation Model

The tilt intonation model, developed by *Taylor* and *Black* [23.20] at the Center for Speech Technology Research of the University of Edinburgh, considers intonation to be a sequence of intonational events, which are parameterized by *tilt parameters*. The model posits four basic types of intonational events: pitch accents, boundary tones, connections (regions in the F_0 contour between two pitch accents, two boundary tones, or a pitch accent and a boundary tone), and silence.

Pitch accents and boundary tones are each modeled by piecewise combinations of quadratic functions; these quadratic functions may be rising or falling. Connections are modeled by straight-line interpolations. The amplitude and duration of the rising and falling quadratic functions, the position of the associated intonational event in the time–F_0 plane, together with a *tilt* value associated with each event constitutes the set of *tilt parameters* associated with the intonational events. The *tilt* parameter represents the amount of rise and fall of each accent. The tilt value is a difference in the amplitudes of the rise and fall functions, divided by their sum [23.21], as shown below:

$$\text{tilt} = \frac{|\text{Amp}_{\text{rise}}| - |\text{Amp}_{\text{fall}}|}{|\text{Amp}_{\text{rise}}| + |\text{Amp}_{\text{fall}}|} \; . \tag{23.5}$$

The tilt value ranges from -1 to 1, where -1 indicates a pure fall, 1 indicates a pure rise and 0 indicates a rise followed by a fall of equal magnitude. Thus, the tilt model uses continuous parameters rather than imposing categorical classification on the intonational events.

Dusterhoff and *Black* [23.21] have shown that the tilt model can be successfully used to predict F_0 contours in a text-to-speech system. The tilt-based F_0 generation process has two stages: a training stage and a testing stage. The training stage requires a training database. The database is labeled with tilt events, either automatically or by hand. For each syllable in the database marked with a tilt event, a set of linguistic-prosodic features are extracted. The features are grouped into separate training sets depending on event type. A CART [23.32] training algorithm is applied to each of the training sets to develop a decision tree for every tilt parameter. The decision trees thus describe the tilt parameters in terms of an optimal subset of the extracted features.

The training stage described above is performed offline. The tilt parameter descriptions obtained in the training stage are used in the testing stage for F_0 contour generation of given text. The given text is labeled with tilt events, and the same set of linguistic-prosodic

features (as in the training set) are extracted. The related tilt parameters are calculated from the extracted features using the descriptions obtained from training. The tilt parameters are then plugged into predetermined quadratic or linear functions to model the pitch accents and boundary tones, or connections respectively.

The IPO Approach

The IPO approach [23.50, 51] was developed at the Institute of Perception Research (IPO) in Eindhoven, the Netherlands in the 1960s. It was originally used to model Dutch intonation. The IPO model is often classified as a *perceptual* intonation model because of the assumptions underlying the model.

1. Not all changes in F_0 are perceived by the human ear.
2. Only F_0 changes that are perceived by the human ear need to be modeled.
3. The human ear perceives tone variations (rise versus fall) and not tone intensities (high versus low).

Given these assumptions, the IPO approach models the raw F_0 contour as a piecewise linear approximation of the original contour, known as a *close copy contour*. It is called a *close copy contour* because, upon resynthesis, it is perceptually indistinguishable from the original F_0 contour. Generating the close copy contour also includes specifying the *declination line* (a line that represents the overall downward trend of the F_0 contour).

Close copy contours are classified into discrete, phonetically defined types of F_0 rises and falls. The classification parameters describe the deviation of the close copy from the declination line, and include descriptive factors such as its height and slope relative to the declination line, its span relative to the span of the declination line, its timing in relation to associated syllable(s) duration, its rate of change, etc. The particular parameters used for classification differ from language to language.

Once an inventory of F_0 rises and falls covering the entire combinatorial space of the classification parameters has been collected, a *grammar* specifying the possible and permissible combinations of the F_0 rises and falls is written in terms of the parameters. When the IPO intonation model is used in speech synthesis systems, this grammar is used for F_0 contour generation. F_0 contours predicted by this grammar must be perceptually equivalent to, and as acceptable as natural F_0 contours [23.19]. The IPO model has been implemented in speech synthesis systems for Dutch [23.52], English [23.53], and German [23.54].

The Fujisaki Intonation Model

The Fujisaki intonation model, presented by Hiroya *Fujisaki* [23.10, 55] is the best known of the class of intonation models known as *superpositional* models (or *overlay* models) of F_0. Superpositional models consider F_0 to be a complicated function that can be decomposed into simpler component functions. In the Fujisaki model, the F_0 contour is considered to be an addition (in the log domain) of two components: the *phrase command* and the *accent command*.

The *phrase command* characterizes the overall trend of the intonation of an utterance, represented by the global movement of the associated F_0 contour. The *accent command* on the other hand, highlights particularly extreme excursions of intonation (to stress certain syllables or words) in the utterance, represented by the local peaks and valleys in the F_0 contour. The phrase command is modeled by pulses, while the accent command is modeled by step functions. The discontinuities in the two commands are then smoothed using separate filters to output phrase and accent components that appear continuous. The phrase and accent components are then added in the log domain to produce an *additive F_0 contour* – the defining characteristic of the Fujisaki model.

To use the Fujisaki model for F_0 prediction in text-to-speech synthesis, the pulses are placed at intonational phrase boundaries, while the step functions are associated with other key phonological units such as *accent groups*. Linguistic and other properties of the text determine the amplitude of each of the commands, and the width of the accent commands. This model has been successfully used for intonation modeling of many languages [23.56, 57].

The Linear Alignment Model

The linear alignment model is another example of a superpositional model. It was developed by *van Santen* and *Möbius* [23.58] at Bell Laboratories. The distinguishing characteristic of this model is that it pays particular attention to the *alignment* between the pitch contour and the segmental stream underlying it.

Its concern with alignment is most effectively expressed in its modeling of the accent curve component. The accent component represents the same aspects of the F_0 contour as the accent component in the Fujisaki model, though it is modeled differently in this model. An accent curve is modeled by parameterized time warps of an accent curve *template*. The *template* can be defined as a sequence of anchor values, $T_p = \langle P_1, P_2, \ldots, P_n \rangle$ that describe the archetypical shape of the associated accent

curve type, P. Also associated with P is an *alignment parameter matrix*, an ensemble of regression weights that describe the alignment of P to the segmental region underlying it. All accent curves of type P have the same template and the same alignment parameter matrix; they differ from each other only in terms of their duration.

Besides the accent curve, two other components of the additive F_0 contour are specified by the linear alignment model: the phrase curve and the *segmental perturbation curve*. As in the Fujisaki model, the phrase curve illustrates the long-term shape of the F_0 contour. The phrase curve is modeled by a piecewise linear function. The *segmental perturbation curve* described the segmental influences on the pitch contour such as pitch increase in vowels following voiceless plosives, and pitch lowering in nasals and glides. The segmental perturbation curves are modeled by exponential decay functions.

The linear alignment Model has been used for generating intonation in the Bell Labs multilingual text-to-speech system [23.59]. To synthesize speech from text, each of the three components of F_0 specified by the linear alignment model have to be related to linguistic entities. The phrase curve is anchored at three points: the start of the utterance, the start of the syllable that carries the nuclear pitch accent, and the end of the utterance. The accent curve is tied to a *left-headed foot*. A *left-headed foot* is defined as a sequence consisting of an accented syllable followed by all unaccented syllables that precede the next accented syllable or a phrase boundary. The degree of emphasis at a particular foot is obtained by multiplying the accent curve by a height factor. The phrase curve is tied to the intonational phrase. Minor and major phrase breaks are distinguished where major phrase breaks occur at sentence ends. Segmental perturbation curves are anchored at vowel onset. The amplitude of this function is determined by the broad class of the onset consonant; it has a maximal value for voiceless consonants, a smaller value for voiced obstruents, and a zero value for sonorants.

The SFC Model

The superposition of functional contours (SFC) model of intonation was developed at the Institute for Speech Communication. It was proposed by *Aubergé* [23.60] and implemented by *Bailly* and *Holm* [23.61]. Like other superpositional models, the principal assumption of the SFC model is that the pitch contour is obtained by a superposition of simpler contours. In case of the SFC model, the simpler contours are multiparametric contours called *functional contours* (FCs).

Functional contours form the core of the distinguishing assumption of this model. They are assumed to directly encode specific *metalinguistic functions* tied to various discourse units, without any intermediate representation [23.62]. *Metalinguistic functions* refer to intonation functions that delimit phonological units and convey propositional and interactional information about these units within the discourse. Examples of metalinguistic functions are hierarchy, segmentation, emphasis, and speaker attitude.

Every functional contour has the following three properties:

1. It is *function-specific*, i. e., tied to a particular metalinguistic function.
2. It spans the extent of the unit(s) tied to the function it encodes – this extent is called the *scope* or domain of the FC.
3. The FC shape is a function of the metalinguistic function it encodes and its scope; however, it is important to note that the FC shape is not specified a priori in this model, rather it emerges in the training phase of the model's implementation.

The SFC model has been implemented for pitch prediction in TTS systems for German, French, and Mandarin Chinese. The metalinguistic functions that will be encoded by the functional contours are defined. One *contour generator* per function is implemented as a neural network. Each contour generator generates a family of functional contours that encode the same metalinguistic function and hence have the same shape, differing only in terms of their time domains. The input to each contour generator is information relating to the scope of the associated function, and the position of each syllable within the scope. The output is four output parameters per syllable (three F_0 values and a lengthening factor).

Training the contour generators to generate a particular pattern of functional contour is not a straightforward process as recovering the unique contributions of the contour generators to their sum (i. e. the F_0 contour) is an ill-posed problem. To determine the individual contributions of each contour generator and the particular pattern of the FC, an *analysis-by-synthesis loop* [23.61] is used.

The Kiel Intonation Model (KIM)

The Kiel intonation model was developed by *Kohler* and his colleagues [23.63, 64] to model intonation patterns in German. In KIM, the F_0 contour is modeled as a sequence of *global intonational units*, each linked to one

emphasized word. The *global intonational units* are considered to be produced and perceived holistically, and cannot be split. These global units are either peaks or valleys or peak-valley combinations, and differ from each other in term of their pragmatic, semantic, syntactic, and meaning functions. They were determined in KIM by means of function-oriented phonetic experiments. KIM postulates that there is a prototypical intonational unit associated with a particular pragmatic–semantic–syntactic function combination. However, KIM does not ignore the *microprosodic phenomena* (e.g., F_0 shifts at the obstruent-vowel boundaries, F_0 changes in nasals and glides; effects of intrinsic pitch are also included in this category) observed in the F_0 contour; it is also incorporated in the model.

Since KIM was developed with a focus on TTS synthesis, the F_0 prediction rules are well specified. KIM applies two sets of rules, namely *symbolic feature rules* and *parametric rules*, for pitch prediction in TTS systems. The symbolic feature rules are applied to *phonological units* which have been annotated with syntactic, pragmatic and semantic markers. The *phonological units* are either segmental (vowels and consonants) or nonsegmental (morphological and phrase boundaries). The symbolic feature rules output the global intonational units associated with the phonological units, encoded as binary features (such as +/−terminal, +/−valley, +/−quest, +/−early, +/−late). These feature values are then used by the parametric rules to generate the F_0 contour of the target utterance. The parametric rules include rules for aligning the global intonational units with the segmental structure of the target utterance, downstepping of accent peaks, speech rate, prosodic boundaries and, finally, articulation-induced microprosody [23.65].

The STEM-ML Intonation Model

Soft template markup language (STEM-ML) is a physiological model of intonation. It was developed by *Shih* and *Kochanski* in 2000 [23.66] to investigate the deviation of Mandarin Chinese tones from their expected canonical shape when occurring in natural sentences. However, this model has been designed to be language independent, and thus can be applied to non-tone languages such as English.

STEM-ML is founded on three key assumptions.

1. Human speech is pre-planned several syllables in advance.
2. Humans produce speech that optimally balances physiological effort required to speak against unambiguity of the spoken message. The speaker expends maximal effort to produce correct prosody at prosodically crucial events because the cost of ambiguity is high at these points. However, he minimizes effort between such events because the cost of ambiguity is low.
3. Speech prosody is continuous and smooth over short time periods.

STEM-ML includes a tagging system (see [23.67] for a complete description of the tag set) for intonation markup and specification, and a quantitative model to generate the F_0 contour. Two important building blocks of the STEM-ML model are parameters and *soft templates*. The parameters are associated with the tags in the tagging system. The *soft templates* are a part of the quantitative intonation generation model. The parameters and the soft templates together generate the F_0 contour.

In this model, the F_0 contour is considered a concatenation of the local accents. The local accents are represented by the *soft templates*. The soft templates are *soft* in the sense that the accent templates allow substantial distortion caused by neighboring accents. The concept of soft templates arise from the previously stated pre-planning assumption. An accent template is affected by past as well as future templates. The degree of distortion is controlled by a parameter called *strength*. The strength parameter reflects the cost of ambiguity in the previously stated assumption regarding optimally balanced speech. So, if strength (hence cost of ambiguity) is large, the template shape remains unchanged to reflect maximal articulatory effort, whereas if it is low, the accent shape is compromised to reflect minimal articulatory effort.

Besides local tags that control local accent shapes, there are global tags that control speaker-specific information. Thus, a STEM-ML model is built on a particular speech corpus. The implementation of the STEM-ML model involves two phases: the learning phase and the generation phase. In the learning phase, the values of the parameters are determined iteratively by minimizing the difference between the actual F_0 of every STEM-ML tagged utterance in the corpus and the F_0 predicted by the model. In the generation phase, when faced with a target utterance, the model first tags the text underlying the utterance, then uses the predetermined values of the parameters associated with the tags to modify the soft templates, and finally, concatenates the modified accent templates to produce the F_0 contour. STEM-ML has been used to model Man-

darin [23.68], Cantonese [23.69], and English [23.70] speech.

Using as-is Prosody in Unit Selection Synthesis

There are currently many unit selection synthesis systems that do not modify the intonation of the concatenated units at all. Some systems such as CHATR use direct duration and F_0 values in their target cost when searching for units, whereas other systems such as the Actor TTS system [23.71] use an abstract feature vector containing information about the location of the unit in the sentence to ensure selection of units that fit the target contour. The main advantage of this approach is that the output speech has a very natural voice quality as no signal modifications are made that degrade the signal. However, data sparsity is unavoidable [23.24,25] and as such there will always be contexts for which no appropriate units can be found, leading to audible prosodic discontinuities at unit boundaries and selection of units with inappropriate prosody that can change the intended meaning of the sentence. In most unit selection systems, the corpus consists of neutral news-style recordings to minimize these prosodic discontinuities at unit boundaries. However, there is an increasing need for expressive speech synthesis, which increases the number of prosodic contexts substantially, making it impossible to record a speech corpus large enough to avoid discontinuities.

Advantages and Disadvantages

As shown in the preceding descriptions, the different intonational models are implemented in TTS systems using different computational mechanisms: linguistic–prosodic rules, templates, neural nets, decision trees, and linear regression. Each of these computational mechanisms have their advantages and disadvantages. The advantages of rule-based approaches are that they are easy to implement and that they produce consistent intonational contours. The disadvantage of the rule-based approaches is that intonational contours lack the richness and variability of natural contours. The machine learning approaches (i. e., those using decision trees, neural nets and linear regression) on the other hand, have the advantage of being able to produce natural-sounding intonational contours, because they learn the mapping between annotated text and corresponding natural pitch contours in the training phase. However, the machine learning approaches have the disadvantage of requiring large amounts of training data to cover the combinatorial space of phoneme sequences and prosodic contexts. Since it is not feasible to cover this entire combinatorial space, machine learning approaches are often beset with data sparsity problems. The reason for capturing the contours related to the same phoneme sequence in different prosodic contexts is that the shape of the contour changes depending on context. One of the main advantages of the template-based approach is its ability to handle the changing shape of an intonational contour depending on context. The template-based approach posits a reasonable number of templates (or prototypical shapes) related to different prosodic events that can be distorted by means of parameters. However, one of the drawbacks of this approach is that the template and parameters may be fit to data in a way that is phonologically absurd.

23.5 Future Approaches

23.5.1 Hybrid Approaches

In order for unit selection synthesis to generate speech with natural and acceptable prosody, a different approach is needed. Some research has been devoted to select natural prosodic contours from dedicated prosodically balanced speech corpora in conjunction with the traditional unit search in a phonetically balanced speech corpus [23.72–74]. This significantly cuts down on the amount of speech data needed, as there is more emphasis on the phonetic context for the unit selection and more emphasis on the prosodic context for the prosody selection. The natural prosody contours are imposed on the units, so a high-quality speech modification algorithm is key for natural sounding speech. The use of dedicated prosody corpora allows for including different speaking styles and affective modes for synthesis. The use of the superpositional model of intonation by *van Santen* et al. [23.74] to select natural phrase and accent curves that are then superimposed to create the final intonation contour, ensures that the resulting contour is smooth and thus there will be no F_0 mismatch at unit boundaries.

23.6 Conclusions

This chapter provided a conceptual framework for prosodic processing in text-to-speech synthesis, and surveys both recent and older approaches. Over the years, many quite diverse approaches have been invented, ranging from approaches that generate prosody by rule to approaches that take prosodic patterns in the database as is. Despite this broad-based attack on this hard problem, there is clearly still much work to be done. Certainly, unit selection systems in which the database provides a very good coverage of a – necessarily restricted – domain will provide the best quality. As we argued, there are practical problems that limit this approach to certain types of applications where custom voices are not needed and where the overall speaking style is fairly muted. We sketched an alternative approach which is data driven, and in fact uses both prosodic patterns and phonetic sequences as found in the data base with a minimum of speech modification. Whether such an approach will be both more practical than, and comparable in quality to, the unit selection-based approach remains to be seen.

References

23.1 J. van Santen: Contextual effects on vowel duration, Speech Commun. **11**(6), 513–546 (1992)

23.2 J. van Santen: Exploring N-way tables with Sums-of-Product models, J. Mathemat. Psychol. **37**(3), 327–371 (1993)

23.3 B. Möbius, J. van Santen: Modeling segmental duration in German text-to-speech synthesis, Proc. 1996 Int. Conf. Spoken Lang. Process. Philadelphia (1996) pp. 2395–2398

23.4 C. Shih, B. Ao: Duration study for the Bell Laboratories Mandarin text-to-speech system. In: *Progress in Speech Synthesis*, ed. by J. van Santen, R. Sproat, J. Olive, J. Hirschberg (Springer, New York 1996) pp. 383–397

23.5 J. van Santen: Assignment of segmental duration in text-to-speech synthesis, Computer Speech Language **8**, 95–128 (1994)

23.6 H. Kato, M. Tsuzaki, Y. Sagisaka: Acceptability for temporal modification of single vowel segments in isolated words, J. Acoust. Soc. Am. **104**(1), 540–549 (1998)

23.7 J. van Santen, C. Shih: Suprasegmental and segmental timing models in Mandarin Chinese and American English, J. Acoust. Soc. Am. **107**(2), 1012–1026 (2000)

23.8 J. van Santen: Segmental duration and speech timing. In: *Computing Prosody*, ed. by Y. Sagisaka, W.N. Campbell, N. Higuchi (Springer, New York 1996)

23.9 J.B. Pierrehumbert: *The phonetics and phonology of English intonation*, Ph.D. Thesis (MIT, Cambridge 1980)

23.10 H. Fujisaki: Dynamic characteristics of voice fundamental frequency in speech and singing. In: *The Production of Speech*, ed. by P.F. MacNeilage (Springer, New York 1983) pp. 39–55

23.11 K. Silverman, M. Beckman, J. Pitrelli, M. Ostendorf, C. Wightman, P. Price, J. Pierrehumbert, J. Hirschberg: ToBI: A standard for labeling English prosody, Proc. 1992 Int. Conf. Spoken Language Processing Banff (1992) pp. 867–870

23.12 K.J. Kohler: Macro and micro F0 in the synthesis of intonation. In: *Papers in Laboratory Phonology I: Between the Grammar and Physics of Speech*, ed. by J. Kingston, M.E. Beckman (Cambridge Univ. Press, New York 1990) pp. 115–138

23.13 M. d'Imperio, D. House: Perception of questions and statements in Neapolitan Italian, Proc. Fifth European Conference on Speech Communication and Technology Rhodes (1997)

23.14 D.J. Broad, F. Clermont: Linear scaling of vowel-formant ensembles (VFEs) in consonantal contexts, Speech Commun. **37**, 175–195 (2002)

23.15 D.H. Klatt: Interaction between two factors that influence vowel duration, J. Acoust. Soc. Am. **54**, 1102–1104 (1973)

23.16 D.H. Klatt: Linguistic uses of segmental duration in English: Acoustic and perceptual evidence, J. Acoust. Soc. Am. **59**, 1209–1221 (1976)

23.17 J. Allen, S. Hunnicut, D. Klatt: *Text-to-Speech: The MITalk System* (Cambridge Univ. Press, Cambridge 1987)

23.18 J.B. Pierrehumbert: Synthesizing intonation, J. Acoust. Soc. Am. **70**, 985–995 (1981)

23.19 R. Sproat (Ed.): *Multilingual Text-to-Speech Synthesis: The Bell Labs Approach* (Kluwer, Dordrecht 1997)

23.20 P. Taylor: Analysis and synthesis of intonation using the Tilt model, J. Acoust. Soc. Am. **107**(3), 1697–1714 (2000)

23.21 K. Dusterhoff, A. Black: Generating F0 contours for speech synthesis using the Tilt intonation theory, Intonation: Theory, Models and Applications, Proc. ESCA Workshop, ed. by A. Botinis, G. Kouroupetroglou, G. Carayiannis (1997) pp. 107–110

23.22 A. Black, P. Taylor: CHATR: A generic speech synthesis system, Proc. COLING'94 Kyoto (1994) pp. 983–986

23.23 A. Black, N. Campbell: Prosody and the selection of source units for concatenative synthesis. In: *Progress in Speech Synthesis*, ed. by J. van Santen, R. Sproat, J. Olive, J. Hirschberg (Springer, New York 1995) pp. 279–292

23.24 J. van Santen: Combinatorial issues in text-to-speech synthesis, Proc. Eurospeech **97**, 2511–2514 (1997)

23.25 B. Möbius: Rare events and closed domains: Two delicate concepts in speech synthesis, Proc. 4rd ESCA Workshop on Speech Synthesis Pitlochry (2001)

23.26 M.D. Riley: Tree-based modeling for speech synthesis. In: *Talking Machines: Theories, Models, and Designs*, ed. by G. Bailly, C. Benoit, T. Sawallis (Elsevier, Amsterdam 1992) pp. 265–273

23.27 D.H. Klatt: Synthesis by rule of segmental durations in English sentences. In: *Frontiers of Speech Communication Research*, ed. by B. Lindblom, S. Öhman (Academic, New York 1979) pp. 287–300

23.28 J.P. Olive, M.Y. Liberman: Text to speech – An overview, J. Acoust. Soc. Am. **78**(Suppl. 1), 6 (1985)

23.29 R. Carlson, B. Granström: A search for durational rules in a real-speech database, Phonetica **43**, 140–154 (1986)

23.30 K.J. Kohler: Zeitstrukturierung in der Sprachsynthese, ITG-Fachbericht **105**, 165–170 (1994), in German

23.31 K. Bartkova, C. Sorin: A model of segmental duration for speech synthesis in French, Speech Commun. **6**, 245–260 (1987)

23.32 L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone: *Classification and Regression Trees* (Wadsworths Brooks, Monterey 1984)

23.33 H. Chung: Duration models and the perceptual evaluation of spoken Korean, Proc. Speech Prosody 2002 Aix-en-Provence (2002)

23.34 R. Batůšek: A duration model for Czech text-to-speech synthesis, Proc. Speech Prosody 2002 Aix-en-Provence (2002)

23.35 N.S. Krishna, H.A. Murthy: Duration modeling of Indian languages Hindi and Telugu, 5th ISCA Workshop of Speech Synthesis Pittsburgh (2005)

23.36 F. Tesser, P. Cosi, C. Drioli, G. Tisato: Prosodic data driven modelling of a narrative style in Festival TTS, Proc. 5th ISCA Workshop on Speech Synthesis Pittsburgh (2005)

23.37 W.N. Campbell: Syllable-based segmental durations. In: *Talking Machines: Theories, Models, and Designs*, ed. by G. Bailly, C. Benoit, T. Sawallis (Elsevier, Amsterdam 1992) pp. 43–60

23.38 E. Klabbers: *Segmental and Prosodic Improvements to Speech Generation*, Ph.D. Thesis (Eindhoven University of Technology, Eindhoven 2000)

23.39 A. Maghbouleh: An empirical comparison of automatic decision tree and linear regression models for vowel durations, Proc. second meeting of the ACL Special Interest Group in Computational Phonology Santa Cruz (1996)

23.40 D.R. Ladd: *Intonational Phonology* (Cambridge Univ. Press, Cambridge 1996)

23.41 J.A. Goldsmith: *Autosegmental and Metrical Phonology* (Blackwell, Oxford 1990)

23.42 N. Campbell, J. Venditti: J-ToBI: An intonation labelling system for Japanese, Proc. Autumn Meeting Acoust. Soc. Jpn. **1**, 317–318 (1995)

23.43 C. Mayo, M. Aylett, D.R. Ladd: Prosodic transcription of Glasgow English: An evaluation study of Glatobi, Proc. ESCA Workshop: Intonation: Theory, Models and Applications, ed. by A. Botinis, G. Kouroupetroglou, G. Carayiannis (ESCA, 1997) pp. 231–234

23.44 M. Reyelt, M. Grice, R. Benzmuller, J. Mayer, A. Batliner: Prosodische Etikettierung des Deutschen mit ToBI. In: *Natural Language and Speech Technology*, ed. by D. Gibbon (Mouton de Gruyter, Berlin 1996) pp. 144–155, in German

23.45 M. Jilka, G. Mohler, G. Dogil: Rules for the generation of ToBI-based American English intonation, Speech Commun. **28**, 83–108 (1999)

23.46 A. Black, A. Hunt: Generating F0 contours from the ToBI labels using linear regression, Proc. 4th Int. Conf. Spoken Language Process. **3**, 1385–1388 (1996)

23.47 C. Traber: F0 generation with a database of natural F0 patterns and with a neural network. In: *Talking Machines: Theories, Models, and Designs*, ed. by G. Bailly, C. Benoit, T. Sawallis (Elsevier, Amsterdam 1992) pp. 287–304

23.48 C. Traber: Syntactic processing and prosody control in the SVOX TTS system for German, Proc. Eurospeech **93**, 2099–2102 (1993)

23.49 C. Traber: *SVOX: The Implementation of a Text-to-Speech System for German*, Ph.D. Thesis (ETH Zurich, Zurich 1995)

23.50 A. Cohen, J. 't Hart: On the anatomy of intonation, Lingua **19**, 177–192 (1967)

23.51 J. 't Hart, R. Collier, A. Cohen: *A Perceptual Study of Intonation: An Experimental-Phonetic Approach to Speech Melody* (Cambridge Univ. Press, Cambridge 1990)

23.52 J. Terken: Synthesizing natural-sounding intonation for Dutch: rules and perceptual evaluation, Computer Speech Language **7**, 27–48 (1993)

23.53 N. Willems, R. Collier, J. 't Hart: A synthesis scheme for British English intonation, J. Acoust. Soc. Am. **84**(4), 1250–1261 (1988)

23.54 J. van Hemert, U. Adriaens-Porzig, L. Adriaens: Speech synthesis in the SPICOS project. In: *Analyse und Synthese gesprochener Sprache*, ed. by H. Tillmann, G. Willee (Georg Olms, Hildesheim 1987) pp. 34–39

23.55 H. Fujisaki, K. Hirose: Modelling the dynamic characteristics of voice fundamental frequency with applications to analysis and synthesis of intonation, Preprints of the Working Group on Intonation, 13th Intl. Congress of Linguists Tokyo (1982) pp. 57–70

23.56 H. Fujisaki: Modelling in the study of tonal features of speech with application to multilingual speech synthesis, Joint Conference of SNLP and Oriental COCOSDA Hua Hin Prachuapkirikhan (2002)

23.57 H. Mixdorff: Quantitative tone and intonation modeling across languages, Proc. Int. Symp. Tonal Aspects of Languages: With Emphasis on Tone Languages Beijing (2004) pp. 137–142

23.58 J. van Santen, B. Möbius: A quantitative model of F_0 generation and alignment. In: *Intonation: Analysis, Modeling and Technology*, ed. by A. Botinis (Kluwer Academic, Dordrecht 1999) pp. 269–288

23.59 J. van Santen, B. Möbius, J. Venditti, C. Shih: Description of the Bell Labs Intonation System, Proc. 3rd ESCA Speech Synthesis Workshop Jenolan Caves (1998) pp. 293–298

23.60 V. Aubergé: Prosody modeling with a dynamic lexicon of intonative forms: Application for text-to-speech synthesis, Proc. ESCA Workshop on Prosody (1993) pp. 62–65

23.61 B. Holm, G. Bailly: Generating prosody by superposing multi-parametric overlapping contours, Proc. Int. Conf. Speech and Language Processing Beijing (2000) pp. 203–206

23.62 G. Bailly, B. Holm: SFC: A trainable prosodic model, Speech Commun. **46**, 364–384 (2005)

23.63 K.J. Kohler: Studies in German intonation, Arbeitsberichte des Instituts für Phonetik und digitale Sprachverarbeitung, Universität Kiel **25**, 295–360 (1991)

23.64 K.J. Kohler: Parametric control of prosodic variables by symbolic input in TTS synthesis. In: *Progress in Speech Synthesis*, ed. by J. van Santen, R. Sproat, J. Olive, J. Hirschberg (Springer, New York 1997) pp. 459–475

23.65 Kohler K.J.: The Kiel Intonation Model (KIM), its Implementation in TTS Synthesis and its Application to the Study of Spontaneous Speech (1995), retrieved on July 15th, 2006 from http://www.ipds.uni-kiel.de/kjk/forschung/kim.en.html

23.66 G.P. Kochanski, C. Shih: Stem-ML: Language independent prosody description, Proc. Int. Conf. Spoken Lang. Process. **3**, 239–242 (2000)

23.67 G.P. Kochanski, C. Shih: Prosody modeling with soft templates, Speech Commun. **39**(3-4), 311–352 (2003)

23.68 G.P. Kochanski, C. Shih: Automated modelling of Chinese intonation in continuous speech, Proc. Eurospeech **01**, 911–914 (2001)

23.69 T. Lee, G. Kochanski, C. Shih, Y. Li: Modeling tones in continuous Cantonese speech, Proc. 2002 International Conference on Spoken Language Processing Denver (2002) pp. 2401–2404

23.70 C. Shih, G. Kochanski: Modeling intonation: Asking for confirmation in English, Proc. 15th Int. Congress of Phonetic Sciences Barcelona (2003)

23.71 S. Quazza, L. Donetti, L. Moisa, P.L. Salza: ACTOR: A multilingual unit-selection speech synthesis system, Proc. 4th ESCA Workshop on Speech Synthesis Pitlochry (2001)

23.72 F. Campillo-Díaz, E.R. Banga: Combined prosody and candidate unit selections for corpus-based text-to-speech systems, Proc. 7th Int. Conference on Spoken Language Processing (2002) pp. 141–144

23.73 A. Raux, A. Black: A unit selection approach to F0 modeling and its application to emphasis, ASRU 2003 St. Thomas (2003)

23.74 J. van Santen, A. Kain, E. Klabbers, T. Mishra: Synthesis of prosody using multi-level sequence units, Speech Commun. **46**(3-4), 365–375 (2005)

20. Rule-Based Speech Synthesis

R. Carlson, B. Granström

In this chapter, we review some of the issues in rule-based synthesis and specifically discuss formant synthesis. Formant synthesis and the theory behind have played an important role in both the scientific progress in understanding how humans talk and also the development of the first speech technology applications. Its flexibility and small *footprint* makes the approach still of interest and a valuable complement to the current dominant methods based on concatenative data-driven synthesis. As already mentioned in the overview by *Schroeter* (Chap. 19) we also see a new trend to combine the rule-based and data-driven approaches. Formant features from a database that can be used both to optimize a rule-based formant synthesis system and to optimize the search for good units in a concatenative system.

20.1 Background

Rule synthesis and formant synthesis are historically highly related. The theory of formant synthesis goes back to the first attempts to model the acoustics of the speech waveform. In the publications by *Fant* [20.1, 2] and *Stevens* [20.3] the foundations of the acoustics of speech and the relation between articulatory models, acoustic manifestations, and spoken communication can be found. The synthesis models were initially evaluated by carefully copying recorded speech. After these initial experiments new methods were developed to control the synthesizer. The classic publications by *Holmes*, *Mattingly*, and *Shearme* [20.4], *Mattingly* [20.5], *Flanagan* [20.6], *Klatt* [20.7], and *Allen*, *Hunnicutt*, and *Klatt* [20.8] present the basic approaches for speech synthesis. An early effort to synthesize speech based on explicit rules can be found in *Carlson* and *Granström* [20.9]. Several review publications on speech synthesis have been published since then: *Sagisaka* [20.10], *Dutoit* [20.11], *Carlson* and *Granström* [20.12]. The paper by *Klatt* [20.13] gives an extensive review of the developments of the speech synthesis technique before the general introduction of concatenative synthesis.

20.2 Terminal Analog

20.2.1 Formant Synthesizers

The traditional text-to-speech systems use a terminal analog as a sound-generating device, i. e., the aim with this kind of synthesizer is only that it should be able to produce the sounds (speech spectra) that are found in natural speech. The internal structure is not a model of acoustic speech production in the vocal tract. The basic concept is the combination of sound sources and filters, describing the transfer function, see *Lawrence* [20.14] and *Fant* [20.15]. The source–filter model exists in several versions; the classical configuration by *Klatt* [20.16], is exemplified in Fig. 20.1. The vocal-tract transfer function is simulated by a sequence of second-order filters in a cascaded arrangement, while a parallel structure is used mostly

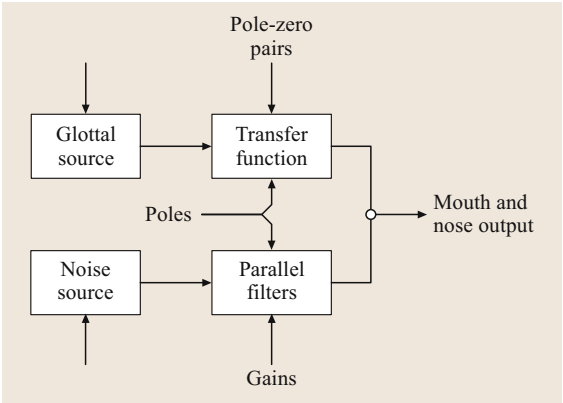


Fig. 20.1 Block diagram of the main components of the Klatt terminal-analog speech synthesizer (after [20.18])

for the synthesis of consonants. One important advantage of a cascade synthesizer is the automatic setting of formant amplitudes. The disadvantage is that it sometimes can be difficult to do detailed spectral matching between natural and synthesized spectra because of the simplified model. Parallel synthesizers such as the one by *Holmes* [20.17] do not have this limitation.

The Klatt model is widely used in research both for general synthesis purposes and for perceptual experiments. A simplified version of this system is used in commercial products that builds on the research at Massachusetts Institute of Technology (MIT): MITalk [20.8], Klattalk [20.20], and DECtalk [20.21].

The formant terminal analog, GLOVE [20.19], based on the OVE synthesizer [20.22], was developed at KTH (Royal Institute of Technology) and has been explored in speech research and several practical applications [20.23, 24]. The structure of the GLOVE synthesizer is shown in Fig. 20.2. The controllable parameters are indicated by two-letter symbols. To the left, the two sound sources can be seen. For mixed excitation the sources are connected in two ways. The parameter N_M flow-modulates the noise source, typical for voiced fricatives. The parameter N_A adds noise to the glottal source, as in breathy or whispered voices. The five parameters above the voice source are the glottal parameters referred to in the voice source section below. The sound source signals are fed into the three parallel branches with poles and zeroes. All are controlled by amplitude parameters (A_N, A_0, A_H and A_C). The upper branch is primarily used for introducing an extra pole and zero in nasals and nasalized sounds. The middle branch is the main branch for sounds produced with glottal excitation and the lowest branch models sounds with supraglottal excitation, such as stops and fricatives. This basic configuration can be augmented in several ways. The interaction between the source and the vocal tract, which can be substantial, is in this case only modeled by the B_M parameter that modulates the bandwidth of the first formant, dependent on the glottal opening, or more precisely the glottal flow. The main difference between the MIT and KTH traditions can be found in how the consonants are modeled. In the OVE case, a fricative is filtered by a zero–pole–pole configuration rather than the parallel branch in the Klatt synthesizer.

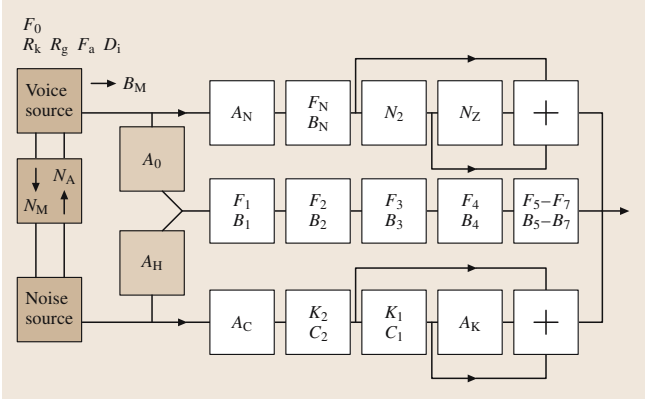


Fig. 20.2 Block diagram of the main components of the terminal-analog speech synthesizer GLOVE (after [20.19])

With terminal analog synthesizers, it is possible to simulate most human voices, and to replicate an utterance without noticeable quality reduction. However, it is interesting to note that some voices are easier to model than others. Despite the progress, speech quality is not good enough in all applications of text to speech. The main reasons for the limited success in formant-based synthesis can be explained by the incomplete knowledge needed for automatic control of the parameters. It should be noted that the transfer of knowledge from phonetics to speech technology and explicit rule-based description has not been an easy process.

20.2.2 Higher-Level Parameters

Since the control of a formant synthesizer can be a very complex task, some efforts have been made to help the developer. The *higher-level parameters* described by *Stevens* and *Bickley* [20.18], *Hanson* and *Stevens* [20.25] and *Stevens* [20.26], for example, explore an intermediate level that is more understand-

able from the developer's point of view compared to the detailed synthesizer specifications. The goal with this approach is to find a synthesis framework to simplify the process and to incorporate the constraints that are known to exist within the process. A formant frequency should not have to be adjusted specifically by the rule developer depending on nasality or glottal opening. This type of adjustment might be better handled automatically according to a well-specified model. The same process should occur with other parameters such as bandwidths and glottal settings. This approach requires detailed understanding of the relation between acoustic and articulatory phonetics. The Prosynth project *Ogden* et al. [20.27] has made use of the multilevel view and included phonetic and prosodic aspects in the approach; see *Heid* and *Hawkins* [20.28].

20.2.3 Voice Source Models

The traditional voice source model has been a simple or double impulse. This is one reason why the voices produced by early text-to-speech systems lack naturalness to a great extent. While the male voice has sometimes been regarded as generally acceptable, an improved glottal source will open the way to more-realistic synthesis of child and female voices and also to greater naturalness and variation in male voices.

Most source models work in the time domain with various controls to manipulate the pulse shape [20.30–40]. One influential voice source model is the LF model [20.41]. It has a truncated exponential sinusoid followed by a variable-cut-off −6 dB/octave low-pass filter modeling the effect of the return phase, i. e., the time from maximum excitation of the vocal tract to complete closure of the vocal folds. Figure 20.3 shows the function of the control parameters. In addition to the amplitude and fundamental frequency control, two parameters largely influence the amplitudes of the two to three lowest harmonics, and one parameter, the high-frequency content of the spectrum. Another vocal source parameter is the diplophonia parameter (inspired by *Klatt* and *Klatt* [20.33]) with which creak, laryngalization or diplophonia can be modelled. This parameter influences the function of the voiced source in such a way that every second pulse is lowered in amplitude and shifted in time.

The acoustic interactions between the glottal source and the vocal tract also have to be considered [20.42]. One of the major factors in this respect is the varying bandwidth of the formants. This is especially true for

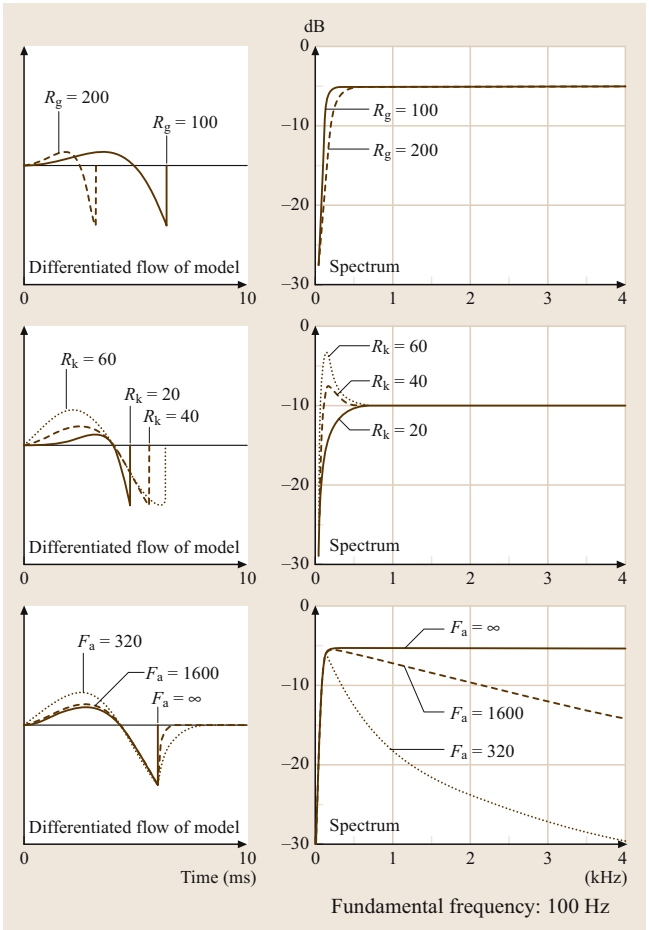


Fig. 20.3 The influence of the parameters R_G, R_K and F_A on the differentiated glottal flow pulse shape and spectrum. The spectra are pre-emphasized by 6 dB/oct (after [20.29])

the first formant, which can be heavily damped during the open phase of the glottal source. However it is not clear that such a variation can be perceived by a listener. Listeners tend to be rather insensitive to bandwidth variation [20.6]. When more-complex models are included, the output from the model has to change from a glottal flow model to a model of the glottal opening. The subglottal cavities can then be included in an articulatory model.

Noise sources have attracted much less research effort than the voiced source. However, some aspects are discussed by *Stevens* [20.3, 43], *Shadle* [20.44], and *Badin* and *Fant* [20.45]. Typically, simple white noise is filtered by resonances that are stationary between each parameter frame. Some formant synthesizers do include

some interaction between the voice source and the noise source, but the interaction is rather primitive. The modeling of transient sounds and aspiration dependent on vocal-cord opening is still a challenge.

20.3 Controlling the Synthesizer

As already mentioned initial work on formant synthesis was related to the development of theories and models of the acoustics of speech. Synthesis experiments were mostly devoted to testing the models and initial speech perception experiments, for example the classic experiments by *Delattre*, *Liberman*, and *Cooper* [20.47]. The work by *Holmes*, *Mattingly* and *Shearme* [20.4] was a major step towards controlling the synthesizer based on symbolic input such as phones or phonemes. Each phoneme was described by a simplified parametric pattern. It was clear that quality was limited if the settings were not adjusted, and rule systems to modify these settings were developed. A number of models were created to take coarticulation into account such as smoothed step functions *Liljencrants* [20.48] or explicit rule descriptions *Klatt* [20.9, 49–51].

Unfortunately rule development for formant synthesis has some problems due to the parametric domain itself. The context-dependent acoustic realizations in terms of formant settings reflect the underlying articulatory gestures rather than the formant pattern in the context (e.g., [20.52]). In a similar way parameters such as variation along the *hyper/hypo-articulation* dimension have a strong influence on the acoustic realization and talking speed [20.53, 54].

A challenging problem in rule development is the locus equations. For example velar stop coarticulation is dependent on many factors and cannot easily be summarized in a single formula [20.55].

These are some of the reasons why formant synthesis has not been competitive in relation to concatenative synthesis. However, the need to synthesize different voices and voice characteristics and to model emotive speech is a strong motivation to keep research on parametric synthesis active. The driving force is that rule-based formant synthesis has the necessary flexibility to model both linguistic and extralinguistic processes. It is, for example, evident that large databases for corpus-based approaches cannot be recorded consistently in a range of desired speaking styles. We will in the following describe some tools that have been used to develop formant synthesis rules and alternative data-driven methods. Rules for prosodic realization will be discussed further in the contribution by Van Santen et al. (Chap. 20).

20.3.1 Rule Compilers for Speech Synthesis

Development tools for text-to-speech systems have received considerable attention. Often the basis of such tools has followed the development of phonological theory. The work on generative phonology and especially the publication of *The Sound Pattern of English* by *Chomsky* and *Halle* [20.56] led to a special kind of synthesis system based on context-sensitive rewrite rules. The linguistically oriented notation inspired speech researchers to create special rule compilers for text-to-speech developments in the 1970s (e.g. [20.36]). The implementations vary depending on the developer's inclination. It is important to note that crucial decisions were often hidden in the systems. The rules may operate rule-by-rule or segment-by-segment. With greater emphasis on prosodic modeling and the related development of nonlinear phonology [20.57], synthesis procedures inspired by such theories were created, as in the systems described by *Hertz* [20.58], *Hertz*, *Kadin*, and *Karplus* [20.59], *Lazzaretto* and *Nebbia* [20.60], *Ceder* and *Lyberg* [20.61], and *Leeuwen* and *Lindert* [20.46, 62]. The common feature of these notations was that they keep information on different linguis-

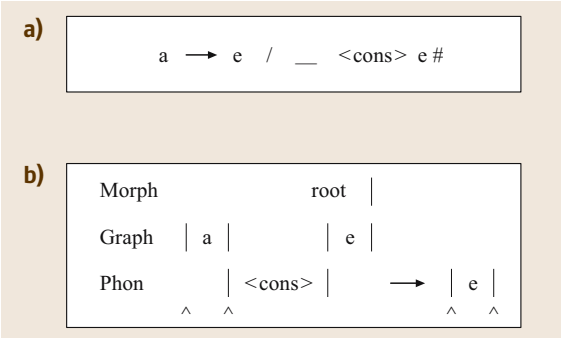


Fig. 20.4 **(a)** Notation according to the KTH *RULSYS* system. (after [20.9]) **(b)** Notation according to the *Speech Maker* system developed at IPO (after [20.46])

tic levels (tiers) separate in a more-explicit way than the essentially linear representation based on generative phonology. This gave potentially higher flexibility, but also more-complex notations. The rules now looked more like two-dimensional schemes rather than one-line representations. This posed greater demands on the developer and his/her tools. Rather than conventional text editors, editors should preferably be graphic editors, where relations between tiers can easily be specified, as in the Speech Maker system developed at IPO (Institute for Preception Research) [20.46]. In Fig. 20.4, this representation is compared to the one used in the KTH system. The rules describe how a letter "a" is pronounced as [e] if it occurs before a single consonant sound and a root final "e". In the KTH notation, the root boundary is introduced as the symbol "#" by earlier rules. After the rule application, the letter origin of the [e] is lost, which places greater demands on the rule order.

20.3.2 Data-Driven Parametric Synthesis

The flexibility in formant synthesis can also be a problem, since articulatory constraints, for example, are not directly included in the formant-based model. The underlying articulatory gestures are not easily transformed to the acoustic domain described by the formant model. When increasing our ambitions to multilingual, multispeaker and multistyle synthesis it is obvious that we want to find at least semiautomatic methods to collect the necessary information, using speech and language databases. Traditionally, speech synthesis has been based on very labor-intensive optimization work. The notion *analysis by synthesis* has not been explored except by manual comparisons between hand-tuned spectral slices and a reference spectrum. The work by *Holmes* and *Pearce* [20.64] is a good example of how to speed up this process. With the help of a synthesis model, the spectra are automatically matched against analyzed speech. Automatic techniques such as this will also probably play an important role in making speaker-dependent adjustments. One advantage of these methods is that the optimization is done in the same framework as that to be used in the production. The synthesizer constraints are thus already imposed in the initial state.

As early as in the 1950s *Peterson* et al. [20.65] suggested that unit concatenation might be a possible solution for speech synthesis. *Dixon* and *Maxey* [20.66] made a special effort to create a unit library for diphone synthesis. Early synthesis research at AT&T based on *diadic units* [20.67] demonstrated an alternative to rule-based formant synthesis.

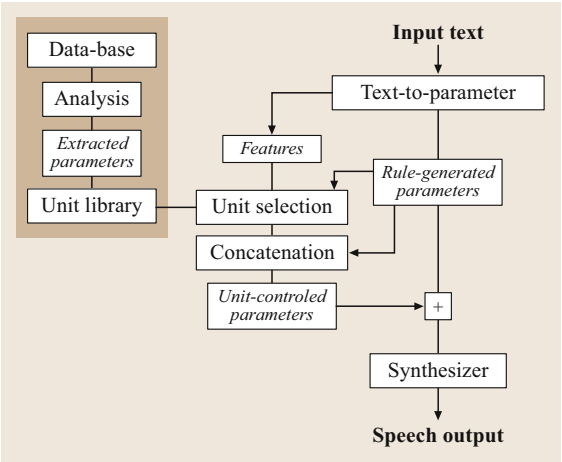


Fig. 20.5 Rule-based synthesis system using a data-driven unit library (after [20.63])

These alternative approaches, which reduce the need for detailed formant synthesis rules but still keep the flexibility of the formant model, extract formant synthesis parameters directly from a labeled corpus. *Mannell* [20.68] reported a promising effort to create a diphone library for formant synthesis. The procedure included a speaker-specific extraction of formant frequencies from a labeled database. In a sequence of papers from Utsunomiya University, Japan automatic formant tracking was used to generate speech synthesis of high quality using formant synthesis and an elaborate voice source (e.g., *Mori* et al. [20.69]). Research efforts to combine data-driven and rule-based methods has recently been reported by *Hertz* [20.70], and *Carlson* and *Granström* [20.63]. The approaches take advantage of the fact that a unit library can better model detailed gestures than general rules. Figure 20.5 illustrates the approach in the KTH text-to-speech system. A database is used to create a unit library and the library information is mixed with the rule-driven parameters. Automatic methods of formant extraction are of course preferred when creating a unit library, due to the amount of data that has to be processed [20.71–73]. However, available methods do not always perform adequately and manual editing is often necessary.

20.4 Special Applications of Rule-Based Parametric Synthesis

In some cases where a speaker model is not readily available database-driven concatenative synthesis is not so well suited. One such example that we have been involved in is the experimental synthesis of varieties of deaf speech in the context of speech training [20.74, 75]. The aim in this project was to synthesize some prototypical versions of the deviations found in the speech of several deaf children, like consonantal realizations, reduced and distorted vowel inventory, and deviant prosody. The rule-based method made it easy to synthesize and evaluate different versions with fewer deviations, thus simulating the effect of (successful) speech training. The often-systematic differences that make intelligibility difficult for occasional listeners, tend to affect the understanding of persons like family less. This points to another use of speech synthesis, where the deviant speech could be *corrected*, but still retain the voice of the deaf or speech-impaired person. One recent example of work aimed at dysarthric speakers is described in *Kain* et al. [20.76], where some improvement was demonstrated both in intelligibility and quality by formant analysis, modification, and resynthesis. The experiment concerned only vowels in nonsense CVC (consonant-vowel-consonant) words uttered by one dysarthric speaker, but points to an interesting possibility.

One early application of speech synthesis was as speech prosthesis for nonvocal individuals. Ideally such a device should be able to use a variety of speaking styles, displaying different attitudes, and emotions in a voice selected by the user. This makes conventional unit-selection methods less suitable, due to the difficulty of recording an appropriate database. Several attempts have made to construct such a prosthesis, e.g., *Murray* et al. [20.77], and *Cudd* et al. [20.78], or the components necessary for, e.g., emotion control [20.79]. However, presently commercially available speech prostheses still lack most of the desired functions.

20.5 Concluding Remarks

Recently we have also seen renewed commercial interest in speech synthesis using the formant model [e.g., the Aurix text-to-speech (TTS) system from 20/20 Speech]. One motivation is the need to generate speech using a very small *footprint*. Thus, one can predict that formant synthesis will again be an important research subject because of its flexibility and also because of how the formant synthesis approach can be compressed into a limited application environment.

References

20.1 G. Fant: *Acoustic Theory of Speech Production* (Mouton, The Hague 1960)

20.2 G. Fant: *Speech Acoustics and Phonetics, Selected Writings Series: Text, Speech and Language Technology*, Vol. 24 (Springer, Berlin, Heidelberg 2006)

20.3 K. Stevens: *Acoustic Phonetics* (MIT Press, Cambridge 1999)

20.4 J. Holmes, I.G. Mattingly, J.N. Shearme: Speech synthesis by rule, Lang. Speech **7**, 127–143 (1964)

20.5 I.G. Mattingly: Synthesis by rule as a tool for phonological research, Lang. Speech **14**(1), 47–56 (1971)

20.6 J.L. Flanagan: *Speech Analysis, Synthesis and Perception* (Springer, Berlin, Heidelberg 1972)

20.7 D. K. Klatt: Structure of a phonological rule component for a synthesis-by-rule program, IEEE Trans. ASSP-**24** (1976)

20.8 J. Allen, M.S. Hunnicutt, D. Klatt: *From Text to Speech. The MITalk System* (Cambridge University Press, Cambridge 1987)

20.9 R. Carlson, B. Granström: A text-to-speech system based entirely on rules, Proc. ICASSP 76, Philadelphia (1976)

20.10 Y. Sagisaka: Speech synthesis from text, IEEE Commun. Mag. **28**(1), 35–41 (1990)

20.11 T. Dutoit: *An Introduction to Text-to-Speech Synthesis* (Kluwer Academic, Dordrecht 1997)

20.12 R. Carlson, B. Granström: *Speech synthesis. In: Hardcastle WJ and Laver J. The Handbook of Phonetic Science* (Blackwell, Oxford 1997) pp. 768–788

20.13 D.K. Klatt: Review of text-to-speech conversion for English, J. Acoust. Soc. Am. **82**(3), 737–793 (1987)

20.14 W. Lawrence: The synthesis of speech from signals which have a low information rate. In: *Commu-*

nication Theory, ed. by W. Jackson (Butterworths, London 1953) pp. 460–469
20.15 G. Fant: Speech Communication Research, Ing. Vetenskaps Akad. Stockholm **24**, 331–337 (1953)
20.16 D.K. Klatt: Software for a cascade/parallel formant synthesizer, J. Acoust. Soc. Am. **67**, 971 (1980)
20.17 J. Holmes: Formant synthesizers, cascade or parallel, Speech Commun. **2**, 251–273 (1983)
20.18 K. Stevens, C. Bickley: Constraints among parameters simplify control of Klatt formant synthesizer, J. Phonetics **19**(1) (1991)
20.19 R. Carlson, B. Granström, I. Karlsson: Experiments with voice modelling in speech synthesis, Speech Commun. **10**, 481–489 (1991)
20.20 D. Klatt: The Klattalk text-to-speech conversion system, Proc. ICASSP **82**, 1589–1592 (1982)
20.21 D. Klatt: DecTalk user's manual, Digital Equipment Corporation (1990)
20.22 J. Liljencrants: The OVE III speech synthesizer, IEEE Trans.Audio Electroac. **16**(1), 137–140 (1968)
20.23 R. Carlson, B. Granström, S. Hunnicutt: A multilanguage text-to-speech module, Proc. ICASSP 82 **82**(3), 1604–1607 (1982)
20.24 R. Carlson, B. Granström, S. Hunnicutt: Multilingual text-to-speech development and applications. In: *Advances in speech, hearing and language processing*, ed. by A.W. Ainsworth (JAI, London 1991)
20.25 H.M. Hanson, K.N. Stevens: A quasiarticulatory approach to controlling acoustic source parameters in a Klatt-type formant synthesizer using HLsyn, J. Acoust. Soc. Am. **112**, 1158–1182 (2002)
20.26 K. Stevens: Toward Formant Synthesis with Articulatory Controls, Proceedings of IEEE Workshop on Speech Synthesis (2002)
20.27 R. Ogden, S. Hawkins, J. House, M. Huckvale, J. Local, P. Carter, J. Dankovicová, S. Heid: ProSynth: An integrated prosodic approach to device-independent natural-sounding speech synthesis, Comput. Speech Lang. **14**, 177–210 (2000)
20.28 S. Heid, S. Hawkins: Synthesizing systematic variation at boundaries between vowels and obstruents. In: *Proceedings of the XIVth International Congress of Phonetic Sciences*, Vol. 1, ed. by J.J. Ohala, Y. Hasegawa, M. Ohala, D. Granville, A.C. Bailey (University of California, Berkeley 1999) pp. 511–514
20.29 C. Gobl, J. Karlsson: Male and female voice source dynamics. In: *Vocal Fold Physiology: Acoustic, Perceptual, and Physiological Aspects of Voice Mechanisms*, ed. by J. Gauffin, B. Hammarberg (Singular, San Diego 1991)
20.30 T. V. Ananthapadmanabha: Acoustic analysis of voice source dynamics, STL-QPSR **2**(3) 1–24 (1984)
20.31 P. Hedelin: A glottal LPC-vocoder, Proc. IEEE, San Diego, 1.6.1–1.6.4 (1984)
20.32 J. Holmes: Influence of the glottal waveform on the naturalness of speech from a parallel formant synthesizer, IEEE Trans. Audio Electroac. **AU-21**, 298–305 (1973)
20.33 D.K. Klatt, L. Klatt: Analysis, synthesis, and perception of voice quality variations among female and male talkers, J. Acoust. Soc. Am. **87**, 820–857 (1990)
20.34 A.E. Rosenberg: Effect of glottal pulse shape on the quality of natural vowels, J. Acoust. Soc. Am. **53**, 1632–1645 (1971)
20.35 M. Rothenberg, R. Carlson, B. Granström, J. Lindqvist-Gauffin: *A three-parameter voice source for speech synthesis", Proc. of Speech Communication Seminar, Stockholm 1974; in Speech Communication*, Vol. 2 (Almqvist and Wiksell, Stockholm 1975) pp. 235–243
20.36 R. Carlson, B. Granström: A Phonetically Oriented Programming Language for Rule Description of Speech. In: *Speech Communication*, Vol. 2, ed. by G. Fant (Almqvist Wiksell, Uppsala 1975) pp. 245–253
20.37 P. Alku: *An automatic inverse filtering method for the analysis of glottal waveforms, Dissertation* (Helsinki University of Technology, Helsinki 1992)
20.38 C. Gobl, A. Ní Chasaide: Acoustic characteristics of voice quality, Speech Commun. **11**, 481–490 (1992)
20.39 C. Gobl, A. Ní Chasaide: The role of voice quality in communicating emotion, mood and attitude, Speech Commun. **40**, 189–212 (2003)
20.40 I. Karlsson: Modelling speaking styles in female speech synthesis, Speech Commun. **11**, 491–497 (1992)
20.41 G. Fant, J. Liljencrants, Q. Lin: A four parameter model of glottal flow, Speech Transmission Laboratory Quarterly and Status Report STL-QPSR No 4 (1985)
20.42 C. Bickley, K. Stevens: Effects of the vocal tract constriction on the glottal source: Experimental and modelling studies, J. Phon. **14**, 373–382 (1986)
20.43 K.N. Stevens: Airflow and turbulence noise for fricative and stop consonants: Static considerations, J. Acoust. Soc. Am. **50**(4), 1180–1192 (1971)
20.44 C.H. Shadle: The Aerodynamics of Speech. Handbook of Phonetics, ed. by W.J. Hardcastle, J. Laver, (Blackwell, Oxford 1997) pp. 33–64
20.45 P. Badin, G. Fant: Fricative modeling: some essentials, Proc. Europ. Conf. Speech Technol. Paris (1989)
20.46 H.C. van Leeuwen, E. te Lindert: Speech Maker: A flexible and general framework for text-to-speech synthesis, and its application to Dutch, Comput. Speech Lang. **7**(2), 149–168 (1993)
20.47 P.C. Delattre, A.M. Liberman, F.S. Cooper: Acoustic loci and transitional cues for consonants, J. Acoust. Soc. Am. **27**, 769–773 (1955)
20.48 J. Liljencrants: Speech synthesizer control by smoothed step functions, STL-QPSR **1969**(4), 43–50 (1969)

20.49 D.H. Klatt: Synthesis of stop consonants in initial position, J. Acoust. Soc. Am. Suppl. **147**, S93 (1970)

20.50 N. Umeda: Linguistic rules for text-to-speech synthesis, Proc. IEEE **64**(4), 443–451 (1976)

20.51 D.K. Klatt: Synthesis by rule of segmental durations in English sentences. In: *Frontiers in Speech Communication Research*, ed. by B. Lindblom, S. öhman (Academic, New York 1979)

20.52 G. Bailly, R. Laboissière, J. L. Schwartz: Formant trajectories as audible gestures: an alternative for speech synthesis, J. Phon. **19**(1), 9–23 (1991)

20.53 B. Lindblom: Explaining phonetic variation: A sketch of the H and H theory. In: *Speech Production Modeling*, ed. by Hardcastle, Marchal (Kluwer Academic, Dordrecht 1990)

20.54 R. J. J. H. van Son, L. Pols: Comparing formant movements in fast and normal rate speech, Proc. Europ. Conf. on Speech Commun. Technol. **89** (1989)

20.55 A. Slater, S. Hawkins: Effects of stress and vowel context on velar stops in British English, ICSLP 92 (Proc. 1992 Int. Conf. Spoken Language Processing) **1**, 57–60 (1992)

20.56 N. Chomsky, M. Halle: *Sound pattern of English* (Harper and Row, New York 1968)

20.57 J.B. Pierrehumbert: *The Phonetics of English Intonation* (IULC, Bloomington 1987)

20.58 S. R. Hertz: Streams, phones, and transitions: toward a new phonological and phonetic model of formant timing, J. Phon. **19**(1) (1991)

20.59 S.R. Hertz, J. Kadin, K.J. Karplus: The Delta rule development system for speech synthesis from text, Proc. IEEE **73**(11), 1589–1601 (1985)

20.60 S. Lazzaretto, L. Nebbia: SCYLA: Speech compiler for your language, Proc. European Conf on Speech Comm and Technology, Edinburgh **1**, 381–384 (1987)

20.61 K. Ceder, B. Lyberg: Yet another rule compiler for text-to-speech conversion? Proc. ICSLP92, Banff, Canada, pp. 1151–1154 (1992)

20.62 H. C. van Leeuwen, E. te Lindert: Speechmaker, text-to-speech synthesis based on a multilevel, synchronized data structure, Proc. ICASSP-**91** (1991)

20.63 R. Carlson, B. Granström: Data-driven multimodal synthesis, Issues Speech Commun. **47**(1–2), 182–193 (2005)

20.64 W. J. Holmes, D. J. B. Pearce: Automatic derivation of segment models for synthesis-by-rule. Proc ESCA Workshop on Speech Synthesis, Autrans, France (1990)

20.65 G. Peterson, W. Wang, E. Sivertsen: Segmentation techniques in speech synthesis, J. Acoust. Soc. Am. **32**, 639–703 (1958)

20.66 N.R. Dixon, H.D. Maxey: Terminal Analog Synthesis of Continuous Speech Using the Diphone Method of Segment Assembly, IEEE Trans. Audio Electroac. **AU-16**, 40–50 (1968)

20.67 J.P. Olive: Rule synthesis of speech from dyadic units, Proc. ICASSP **77**, 568–570 (1977)

20.68 R. H. Mannell: Formant diphone parameter extraction utilising a labeled single speaker database. In: Proc. ICSLP-98 (1998)

20.69 H. Mori, T. Ohtsuka, H. Kasuya: A data-driven approach to source-formant type text-to-speech system, ICSLP **2002**, 2365–2368 (2002)

20.70 S. Hertz: Integration of Rule-Based Formant Synthesis and Waveform Concatenation: A Hybrid Approach to Text-to-Speech Synthesis, In: Proc. IEEE 2002 Workshop on Speech Synthesis, 11–13, Santa Monica (2002)

20.71 D. Talkin: Looking at Speech. In: Speech Technology, 74–77 (1989)

20.72 A. Acero: Formant analysis and synthesis using hidden Markov models, Proc. Eurospeech **99**, 1047–1050 (1999)

20.73 M. Lee, J. van Santen, B. Möbius, J. Olive: Formant tracking using context-dependent phonemic information, IEEE TSAP **13**(5), 741–750 (2005)

20.74 A.-M. Öster: The use of a synthesis-by-rule system in a study of deaf speech, STL-QPSR 1/ **1985**, 95–107 (1985)

20.75 B. Granström, A.-M. Öster: Speech synthesis for hearing impaired persons – in research, training and communication, STL/QPSR 2-3/ **94**, 93–111 (1994)

20.76 A. Kain, X. Niu, J. Hosom, J. Miao, J. van Santen: Formant Re-synthesis of Dysarthric Speech. Proceedings of IEEE Workshop on Speech Synthesis (2004)

20.77 I. Murray, J. Arnott, N. Alm, A. Newell: A communication system for the disabled with emotional synthetic speech produced by rule, Procs. Eurospeech **91**(1), 311–314 (1991)

20.78 P. A. Cudd, S. Hunnicutt, J. Arthur, B. Granström, S. Aguilera, B. Waernulf, P. Dalsgaard, G. Wilson: Voices, attitudes and emotions in speech synthesis. In Placencia Porrero, I., and Puig de la Bellacasa, P. (Eds.), *Proc of 2nd TIDE Congress on The European Context for Assistive Technology* (pp. 344-347). Paris, Amsterdam: IOS Press Ohmsha (1995)

20.79 J. Cahn: The generation of affect in synthesized speech, J. Am. Voice I/O Soc. **8** (1990)

24. Voice Transformation

Y. Stylianou

Voice transformation refers to the various modifications one may apply to the sound produced by a person, speaking or singing. In this chapter we give a description of various ways in which one can modify a voice and provide details on how to implement these modifications using a simple, but quite efficient, parametric model based on a harmonic representation of speech. By discussing the quality issues of current voice transformation algorithms in conjunction with the properties of speech production and perception systems we try to pave the way for more-natural voice transformation algorithms in the future.

24.1 Background

Voice transformation refers to the various modifications one may apply to the sound produced by a person, speaking or singing. Voice transformation involves signal processing and the physics (or at least the understanding) of the speech production process and natural language processing. Driven mainly by its applications, signal processing has evolved faster than the physics of speech processing, even giving the impression that signal processing alone may be required to achieve high-quality voice transformation. To an external observer, this is similar to the problem of how to make an omelette without eggs. It is not surprising therefore that, although for some categories of voice transformation good quality of speech is produced, this is not true in general. While it is relatively easy to explain to a non-speech expert the necessity of speech modeling by providing examples from the history of telecommunications, this is not obvious for voice transformation. About two decades ago, it was easy to explain the applications of voice transformation technology to a speech processing engineer by providing examples from a specific area of speech technology: concatenating speech synthesis. In the late 2000s, providing a reason for voice transformation to both a speech expert and nonexpert faced the same difficulty. One reason for this was that the main application of voice transformation, that of concatenating speech synthesis, had evolved in a direction where it seemed that signal processing was no longer needed for this application. Such a point of view, however, was also supported by the quality problems perceived in a modified speech signal.

Recently, interest in voice transformation has increased substantially, and it is again the application of speech synthesis that is setting the pace. Voice

transformation is a flexible, possibly simple, and efficient way to produce the variety needed in the current text-to-speech (TTS) systems based on the concatenation of units (both large and small) [24.1]. In this chapter, we give a description of various ways in which one can modify a voice and provide details of how to implement these modifications using a simple, but quite efficient, parametric model based on a harmonic representation of speech. Discussing quality issues of current voice transformation algorithms in conjunction with properties of the speech production and perception systems we try to pave the way for more-natural voice transformation algorithms in the future.

24.2 Source–Filter Theory and Harmonic Models

24.2.1 Harmonic Model

When designing voice transformation techniques it is often convenient to refer to the source–filter model of speech production. According to this model, speech is viewed as the result of passing a glottal excitation signal (source) through a time-varying linear filter that models the resonant characteristics of the vocal tract. The most well-known source–filter system is that based on linear prediction (LP) of speech [24.2]. In its simplest form, a time-varying filter modeled as an autoregressive (AR) filter is excited by either quasiperiodic pulses (during voiced speech), or noise (during unvoiced speech). Many attempts have been made to improve the source (excitation) signal in the LP context. This includes multipulse LP [24.3], and code-excited linear prediction (CELP) [24.4]. A more-compact and at the same time flexible representation of the excitation signal has been proposed from a family of speech representations referred to as sinusoidal models (SM) [24.5]. In SM, the excitation signal for both voiced and unvoiced speech frames is represented by a sum of sinusoids:

$$e(t) = \sum_{k=0}^{K(t)} a_k(t) \mathrm{e}^{\mathrm{i}\phi_k(t)} \,, \tag{24.1}$$

where $a_k(t)$ and $\phi_k(t)$ are the instantaneous excitation amplitude and phase of the k-th sinusoid, respectively, and $K(t)$ is the number of sinusoids, which may vary in time. Especially for speech signals, a model where the sinusoids are harmonically related is quite valid (in the mean-squared-error (MSE) sense) while it allows a simple and convenient way of applying various modifications to the speech signal. In this case:

$$\dot{\phi}_k(t) = 2\pi k f_0(t) \,, \tag{24.2}$$

where $f_0(t)$ is the instantaneous fundamental frequency, which will also be referred to as the *pitch* in this chapter. Such a representation is still valid for both voiced and unvoiced speech frames. In the case of unvoiced speech frames a constant fundamental frequency is considered (i. e., 100 Hz) resulting in a Karhunen–Loeve representation of this speech category [24.5]. A further simplification of the excitation signal is convenient assuming that the excitation amplitude, $a_k(t)$, is constant over time and equal to unity: $a_k(t) = 1$. Based on these simplifications, the time-varying linear filter that models the resonant characteristics of the vocal tract approximates the combined effect of:

1. the transmission characteristics of the supraglottal cavities (including radiation at the mouth opening)
2. the glottal pulse shape

Its time-varying transfer function can be written

$$H(f;t) = G(f;t)\mathrm{e}^{\mathrm{i}\Psi(f;t)} \,, \tag{24.3}$$

where $G(f;t)$ and $\Psi(f;t)$ are, respectively, referred to as the time-varying amplitude and phase of the system. Speech processing is often (if not always) performed in a frame-by-frame basis, where each frame (i. e., about 20 ms) is considered to be a stationary process. In this case, inside a frame, the filter $H(f;t)$ is considered as linear time invariant (LTI). Then, the output speech signal $s(t)$ can be viewed as the convolution of the impulse response of the LTI filter, $h(t)$, and the excitation signal, $e(t)$:

$$s(t) = \int_0^t h(t-\tau)e(\tau)\mathrm{d}\tau \,. \tag{24.4}$$

Recognizing then that the excitation signal is just the sum of $K(t)$ eigenfunctions of the filter, $H(f)$, the

following speech model is obtained:

$$s(t) = \sum_{k=0}^{K(t)} G[f_k(t)]\, \mathrm{e}^{\mathrm{i}\{\phi_k(t)+\Psi[f_k(t)]\}} = \sum_{k=0}^{K(t)} A_k(t)\, \mathrm{e}^{\mathrm{i}\theta_k(t)}\,, \tag{24.5}$$

where $f_k(t) = k f_0(t)$ (the eigenfrequencies). The harmonic amplitude $A_k(t)$ of the k-th harmonic is the system amplitude $G[f_k(t)]$ (the eigenvalue). The phase $\theta_k(t)$ of the k-th harmonic is the sum of the excitation phase $\phi_k(t)$ and the system phase $\Psi[f_k(t)]$:

$$\theta_k(t) = \phi_k(t) + \Psi[f_k(t)]\,; \tag{24.6}$$

$\theta_k(t)$ is often referred to as the *instantaneous phase* of the k-th harmonic.

24.2.2 Analysis Based on the Harmonic Model

Parameters of the harmonic model of speech may be estimated by minimizing a least-squares criterion [24.6] or by minimizing a mean-squared error that leads to a simple *peak-picking* approach [24.5]. The peak-picking approach results in a sinusoidal rather than a harmonic model. A second step is then required to fit a harmonic model to this sinusoidal model by selecting the fundamental frequency that best represents the set of estimated sinusoids. At each analysis time instant, t_a^i, a set of parameters are estimated: the fundamental frequency, f_0^i, the harmonic amplitudes, A_k^i, and the harmonic phases, θ_k^i.

Use of the harmonic model for voice transformations is simplified if the distance between two successive analysis time instants is equal to the local pitch period, $P(t_\mathrm{a}^i) = 2\pi / f_0^i$:

$$t_\mathrm{a}^{i+1} = t_\mathrm{a}^i + P\left(t_\mathrm{a}^i\right)\,. \tag{24.7}$$

Another important step before synthesis is required: the estimation of the *amplitude and phase envelopes*, $A(f)$ and $\theta(f)$ (i. e., a continuous function of frequency) from the discrete set of amplitude and phase values, respectively.

While a number of methods can be used to estimate the amplitude envelope, for example, the linear prediction and homomorphic estimation techniques [24.7], it is desirable to use a method that yields an envelope that passes through the measured harmonic amplitudes. Such a technique was developed for the spectral envelope estimation vocoder (SEEVOC) [24.8] and was used in the sinusoidal model in [24.5]. Another approach was proposed in [24.9]: it provides a continuous frequency envelope when values of this envelope specified only at discrete frequencies (i. e., exactly the situation in the previously described harmonic representation). This approach makes use of cepstral coefficients and is based on a frequency-domain least-squares criterion combined with regularization to increase estimation robustness.

For the phase envelope $\theta(f)$, the previous techniques cannot be used since the phase values have been estimated modulo 2π (principal values). Therefore, a phase unwrapping algorithm has to be used. Two main approaches exist:

1. phase continuity by adding appropriate multiples of 2π to the principal phase values [24.10]
2. continuity by integration of the phase derivative

These algorithms try to obtain a continuous phase envelope in the frequency domain. An extension of these techniques to preserve the continuity in the time domain as well has been proposed using the information of phase from previous voiced frames [24.11].

An alternative to the phase envelope approach is the use of a minimum phase model for the system phase, while for the excitation phase a representation of the excitation in terms of its impulse locations (*onset times*) is used [24.12]. This approach, however, lacks robustness because estimation of the onset times requires precision that is not always easy to obtain.

Next, we will consider the case when a spectral envelope, $A(f)$, and phase envelope, $\theta(f)$, are provided.

24.2.3 Synthesis Based on the Harmonic Model

Without speech modification, synthesis time instants, t_s^i coincide with the analysis time instants t_a^i, i.e., $t_\mathrm{s}^i = t_\mathrm{a}^i$, $\forall i$.

Let $(A_k^i, \theta_k^i, f_0^i)$ and $(A_k^{i+1}, \theta_k^{i+1}, f_0^{i+1})$ denote the set of parameters at synthesis time instant t_s^i and t_s^{i+1} for the k-th harmonic, respectively. Amplitudes and phases are obtained by sampling the phase and amplitude (spectral) envelopes at the harmonics of the fundamental frequencies f_0^i and f_0^{i+1}. The instantaneous amplitude $A_k(t)$ is then obtained by linear interpolation of the

estimated amplitudes at the frame boundaries:

$$A_k(t) = A_k^i + \frac{A_k^{i+1} - A_k^{i+1}}{t_s^{i+1} - t_s^i} t \quad \text{for } t_s^i \le t < t_s^{i+1} . \tag{24.8}$$

In contrast to the third-order polynomial used in [24.13, 14], the harmonic model allows the use of a simple first-degree polynomial for the phase. First, the phase at t_s^{i+1} is predicted from the estimated phase at t_s^i by

$$\hat{\theta}_k^{i+1} = \theta_k^i + k2\pi f_{0\text{av}}\left(t_s^{i+1} - t_s^i\right) , \tag{24.9}$$

where $f_{0\text{av}}$ is the average value of the fundamental frequencies at t_s^i and t_s^{i+1}:

$$f_{0\text{av}} = \frac{f_0^i + f_0^{i+1}}{2} . \tag{24.10}$$

Next, the phase θ_k^{i+1} is augmented by the term $2\pi M_k$ (M_k is an integer) in order to approach the predicted value. Therefore, the value of M_k is given by

$$M_k = \left\langle \frac{1}{2\pi}\left(\hat{\theta}_k^{i+1} - \theta_k^{i+1}\right) \right\rangle . \tag{24.11}$$

Then, the instantaneous phase $\theta_k(t)$ is simply obtained by linear interpolation

$$\theta_k(t) = \theta_k^i + \frac{\theta_k^{i+1} + 2\pi M_k - \theta_k^i}{t_s^{i+1} - t_s^i} t , \quad t_s^i \le t < t_s^{i+1} . \tag{24.12}$$

Having determined the instantaneous values of the harmonic amplitudes and phases the estimated speech signal (a harmonic representation of the speech signal) is then obtained by:

$$\hat{s}(t) = \sum_{k=0}^{K} A_k(t) \cos\left[\theta_k(t)\right] , \tag{24.13}$$

where $A_k(t)$ is given by (24.8) and $\theta_k(t)$ by (24.12).

Based on the source–filter model various speech modification methods can now be defined. Some of these refer only to the source signal, others only to the filter, while others apply to both the source and filter. Moreover, by developing the source–filter model in the context of the harmonic representation of speech signals, a mathematical notation regarding these modifications can be introduced that will be used throughout the rest of the chapter.

24.3 Definitions

24.3.1 Source Modifications

Modifications in the source signal are usually referred to as *prosodic modifications* and include three main types: time-scale modification, pitch modification, and intensity modification.

Time-Scale Modification

The goal of time-scale modification is to change the apparent rate of articulation without affecting the perceptual quality of the original speech. This requires the pitch contour to be stretched or compressed in time, and the formant structure to be changed at a slower or faster rate than the rate of the input speech, but otherwise not be modified. Figure 24.1 shows an example of time stretching where the pitch period contour is slowed down but not modified.

Pitch Modification

The goal of pitch modification is to alter the fundamental frequency in order to compress or expand the spacing between the harmonic components in the spectrum while preserving the short-time spectral envelope (the locations and bandwidths of the formants) as well as the time evolution. In contrast to time-scale modifica-

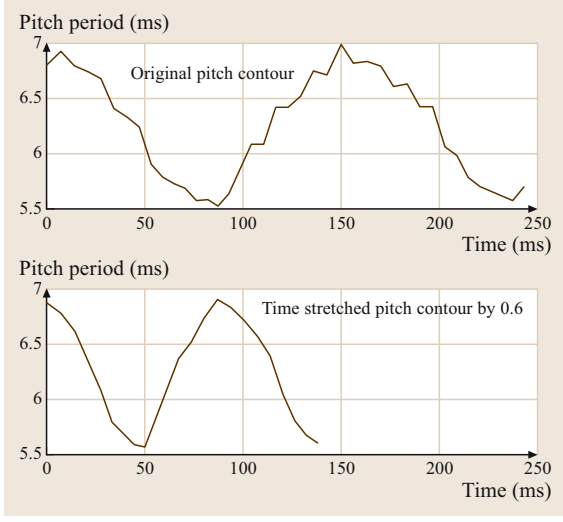


Fig. 24.1 Pitch-period contour: original and time-stretched by a factor of 0.6

tions, in this case the pitch contour is modified *without* modifying, however, the time resolution of the pitch contour. Figure 24.2 shows an example of pitch modification by constant pitch-scale factor (0.6): the time evolution is preserved and the pitch-period contour is scaled by 0.6. In this case, the input fundamental frequency is increased by a factor of $1/0.6$. This could give the impression that a male voice will sound more like a female voice, while a female voice will more sound like a child's voice.

Intensity Modification

It is widely considered that intensity modification is the simplest modification among the prosodic modifications. This is because it can be easily performed by associating an intensity scale factor at each analysis time instant of a signal. The signal is then just multiplied by this scale factor. In the case of a parametric model like the harmonic model developed previously, the scale factor is applied to the harmonic amplitudes $A_k(t)$ in (24.5). It may seem strange to modify a prosodic feature by changing a parameter corresponding to the vocal-tract filter. However, it should be remembered that the filter has been considered as an LTI filter; therefore, multiplying the amplitude of the excitation signal by a constant results in multiplying the harmonic amplitudes, $A_k(t)$, by the constant.

24.3.2 Filter Modifications

By filter modification we mean the modification of the magnitude spectrum of the frequency response of the vocal tract system, $|H(\omega)|$. It is widely accepted that $|H(\omega)|$ carries information of speaker individuality. Representations of the magnitude spectrum (i. e., mel frequency cepstral coefficients (MFCC), line spectrum frequencies (LSF), etc.) have been used a lot in the area of speaker identification and recognition as well as for speaker normalization for robust speech recognition. Therefore, by modifying the magnitude spectrum of the vocal tract, speaker identity may be controlled. We may distinguish two types of filter modification, which are described below.

Without a Specific Target

In this case, the filter characteristics of a speaker are modified in a general way without having a specific target (speaker) in mind. For example, we may wish to modify the overall quality of a speech signal produced by a female speaker so that it sounds as if it had been produced by an older female speaker. Based on the source–filter theory for the production of speech

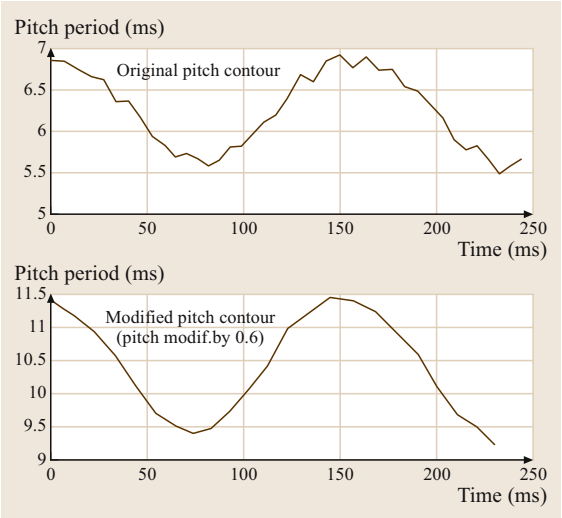


Fig. 24.2 Pitch-period contour: original and modified by applying a pitch modification factor of 0.6

we know that the formants for a female voice are distributed in higher frequencies than the formants from an older female voice. Similar observations are valid for the harmonic frequencies. Therefore, one can modify in a general way the power spectrum of a speaker so that the resulting spectrum has the characteristics of a family of speakers (child's, old person's voices, etc.). Figure 24.3 shows an example of filter modification to modify the spectrum from a female voice to a spectrum *similar* to an old female person. For this one should compress the

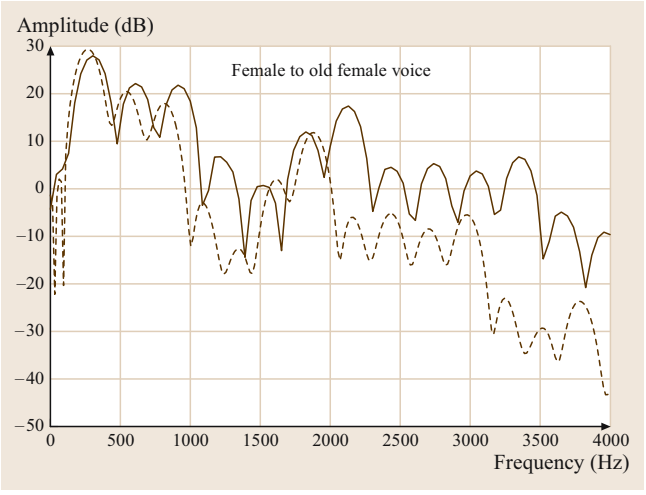


Fig. 24.3 Female (*solid line*) to an old female (*dashed line*) filter modification

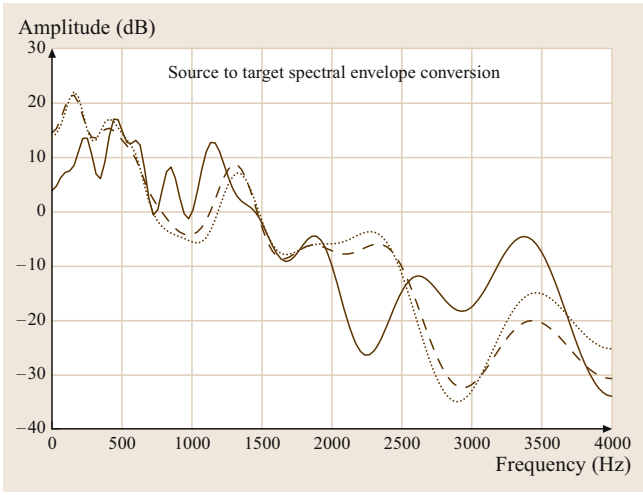


Fig. 24.4 Source (*solid line*) to target (*dashed line*) filter transformation. The transformed filter is shown by a *dashed–dot line*

frequency information so that formants and harmonics are moved towards lower frequencies. The result shown in Fig. 24.3 was obtained by combining two operations: lowering the sampling frequency of the source signal (female voice, 44 100 Hz) to 18 000 Hz, and then applying time-scale modification with a factor of 8/9. The result signal can be played back at 16 000 Hz without any modification in the articulation rhythm. The quality of the modified signal is similar to that the initial signal.

With a Specific Target

In this case, the filter of a speaker (the source speaker) is modified in such a way that the modified filter approximates in the mean-squared sense the characteristics of the filter of another speaker (the target speaker). Usually we refer to this type of modification as a *transformation* or *conversion*. An example of a such transformation is depicted in Fig. 24.4. In this example the original spectrum is shown by a solid line, the target spectrum by a dashed line, and the transformed original to target spectrum by a dashed–dot line. To obtain the transformed spectrum, there is a learning process using many similar examples of the source and the target spectrum; therefore, the transformed spectrum is equal to the average spectrum of the target spectra used during the training process. Details about this type of spectral transformation will be provided in the next section.

24.3.3 Combining Source and Filter Modifications

To transform a female into a male voice, performing filter modification or only pitch modification alone may not provide convincing results. In most cases, source and filter modifications must be combined. For example, the prosody characteristics of a speaker may be a critical cue used for the identification of the speaker by others (speaking style) while at the same time, vocal-tract characteristics are also important for identification. Therefore, if we want to modify the voice of the speaker so that it sounds like the voice of another speaker, prosody and vocal tract modifications should be combined. If a target speaker is provided then this combined source and filter modifications is referred to as *voice conversion* or *transformation*. In contrast, when a specific target is not provided, this is usually referred to as *voice modification*.

Voice morphing is another type of combined source and/or filter modifications. In this case the same two sentences are uttered by two speakers and then a third speaker may be generated having characteristics from both speakers. This is mainly achieved by a dynamic time warping (DTW) algorithm between the two sentences, aligning the acoustic data and then applying a linear or other type of interpolation between the aligned data (source and/or filter characteristics). Sometimes this type of voice transformation is confused with voice conversion. Note that in voice conversion the sentence to be converted, uttered by the source speaker, *has never been uttered* by the target speaker. However, in voice morphing there are two source speakers that generate a new voice saying the same text as the two source voices. In voice conversion, there is only one source speaker and one target speaker, and the voice characteristics of the source speaker should be transformed into the voice characteristics of the target speaker (i. e., a new speaker is not generated in this case).

In the next section we will provide details about the main prosodic (time and pitch) modifications and the filter modifications. Then a system for voice conversion will be presented.

24.4 Source Modifications

Pitch synchronous analysis is the key to the simplicity of many source (prosodic) modification algorithms and is defined as follows. Given an analysis time instant, t_a^i, the next analysis time instant, t_a^{i+1} is determined by the

local pitch period at t_a^i, $P(t_a^i)$, using (24.7). The length of the analysis window is proportional to the local pitch period (usually two local pitch periods are used). We may distinguish two types of pitch synchronous analysis. The first one may be referred to as *strict* pitch synchronous analysis, where the analysis time instants are supposed to coincide with the glottal closure instants (GCIs, sometimes called *pitch marks*). In the other, referred to as *relaxed* pitch synchronous analysis, the analysis time instants do not (necessarily) coincide with the GCIs. Since the estimation of pitch marks from the speech signal is not a robust process, resulting sometimes in an incoherent synthesis, relaxed pitch synchronous analysis seems to be easier to use than the fixed approach. However, this is not true. Pitch modification requires the re-estimation of phases. Coherent synthesis mainly means synthesis without linear phase mismatches. Strict pitch synchronous methods explicitly remove any linear phase mismatch between successive frames by using GCIs. Relaxed pitch synchronous methods, however, need to re-estimate the linear phase component for the new pitch values, which is not a trivial task. Phase models [24.12] and estimation of phase envelopes [24.11] try to overcome these problems.

For the system presented here, we will consider that the analysis time instants (Sect. 24.2.2) have been determined in a relaxed pitch synchronous way.

Next, we will see how the pitch synchronous scheme allows the use of simple and flexible techniques for time-scale and pitch-scale modifications. The first step consists of finding out the synthesis time instants t_s^i (or synthesis pitch marks) according to the desired time-scale and pitch-scale modification factors. The modified signal is then obtained by using the new synthesis time instants.

24.4.1 Time-Scale Modification

We recall that the objective of time-scale modification is to alter the apparent rate of articulation without affecting the spectral content: the pitch contour and time evolution of the formant structure should be time scaled, but otherwise not modified [24.15].

From the stream of analysis time instants t_a^i and the desired time-scale modification factor $\beta(t)$, $(\beta(t) > 0)$ the synthesis time instants t_s^i will be determined. The mapping $t_a^i \rightarrow t_s^i = D(t)$ is referred to as the time-scale warping function, which is defined as the integral of $\beta(t)$:

$$D(t) = \int_0^t \beta(\tau)\,\mathrm{d}\tau \ . \tag{24.14}$$

Note that for a constant time modification rate $\beta(t) = \beta$, the time-scale warping function is linear: $D(t) = \beta t$. The case $\beta > 1$ corresponds to slowing down the rate of articulation by means of a time-scale expansion, while the case $\beta < 1$ corresponds to speeding up the rate of articulation by means of a time-scale compression. Thus, speech events that take place at a time t_{or} in the original time scale will occur at a time $t_{mo} = \beta t_{or}$ in the new (modified) time scale.

As an example, let us assume that at each analysis tim -instant t_a^i a time-scale modification factor b_s has been specified. Thus, $\beta(t)$ is a piecewise constant function, i. e., $\beta(t) = \beta_s$, $t_a^i \le t < t_a^{i+1}$. It follows therefore that the time-scale warping function $D(t)$ can then be written

$$D(t) = D(t_a^i) + \beta_s\left(t - t_a^i\right) , \quad t_a^i \le t < t_a^{i+1} \tag{24.15}$$

with $D(t_a^1) = 0$.

Having specified the time-scale warping function $D(t)$, the next step consists of generating the stream of the synthesis time instants t_s^i, *while preserving the pitch contour*: the pitch in the time-scaled signal at time t should be as close as possible to the pitch in the original signal at time $D^{-1}(t)$. In other words, $t \rightarrow P'(t) = P[D^{-1}(t)]$. We now have to find a stream of synthesis pitch marks (synthesis time instants) t_s^i, such that $t_s^{i+1} = t_s^i + P'(t_s^i)$. To solve this problem, the use of a stream of *virtual pitch marks*, t_v^i, in the original signal related to the synthesis pitch-marks by

$$\begin{aligned} t_s^i &= D\left(t_v^i\right) , \\ t_v^i &= D^{-1}\left(t_s^i\right) , \end{aligned} \tag{24.16}$$

is proposed in [24.15]. Assuming that t_s^i and t_v^i are known, we determine t_s^{i+1} (and t_v^{i+1}), such that $t_s^{i+1} - t_s^i$ is approximately equal to the pitch in the original signal at time t_v^i. This can be expressed as

$$t_s^{i+1} - t_s^i = \frac{1}{t_v^{i+1} - t_v^i} \int_{t_v^i}^{t_v^{i+1}} P(t)\,\mathrm{d}t \tag{24.17}$$

with $t_s^{i+1} = D(t_v^{i+1})$. According to this equation, the synthesis pitch period $t_s^{i+1} - t_s^i$ at time t_s^i is equal to the mean value of the pitch in the original signal calculated over the time interval $t_v^{i+1} - t_v^i$. Note that this time interval $t_v^{i+1} - t_v^i$ is mapped to $t_s^{i+1} - t_s^i$ by the mapping function $D(t)$.

The integral equation (24.17) is easily solved because $D(t)$ and $P(t)$ are piecewise linear functions. Figure 24.5 illustrates an example of the computation of synthesis pitch marks for time-scale modification by 1.5.

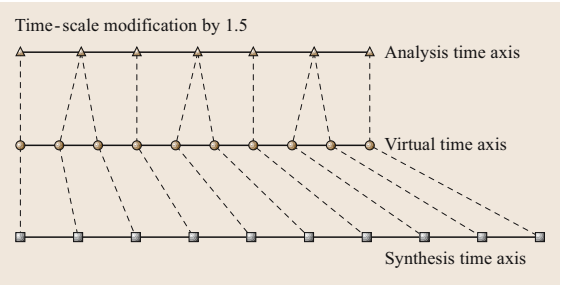


Fig. 24.5 Computation of the synthesis pitch marks for time-scale modification by 1.5

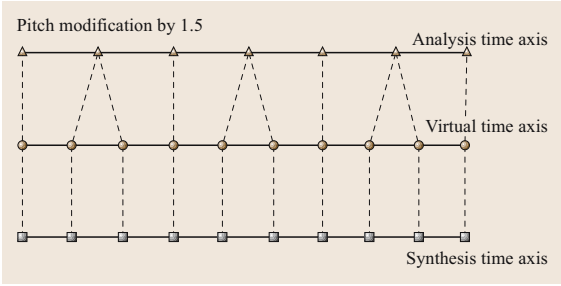


Fig. 24.6 Computation of the synthesis pitch marks for pitch modification by 1.5

24.4.2 Pitch Modification

The goal of the pitch-scale modification is to alter the fundamental frequency of a speaker while the spectral envelope of the speaker's vocal-tract system function is unchanged. Obviously, pitch modification is only applied on the voiced speech frames. The first step consists of computing the synthesis time instants t_s^i from the stream of the analysis time instants t_a^i and the pitch-scale modification factors $a(t)$, with $a(t) > 0$. We recall that the analysis time instants are set in a pitch synchronous way. We require the same for the synthesis time instants: $t_s^{i+1} = t_s^i + P'(t_s^i)$, where $P'(t_s^i)$ is approximately equal to the pitch period in the original signal around time t_a^i scaled by $1/\alpha(t_a^i)$:

$$P'\left(t_s^i\right) = \frac{P\left(t_a^i\right)}{\alpha\left(t_a^i\right)} \,. \tag{24.18}$$

Given the synthesis time instant t_s^i, the next synthesis time instant t_s^{i+1} is obtained by setting the synthesis pitch period to be equal to the mean value of the scaled pitch period (by $1/\alpha(t_a^i)$) in the original signal calculated over the time frame $t_s^{i+1} - t_s^i$:

$$t_s^{i+1} - t_s^i = \frac{1}{t_s^{i+1} - t_s^i} \int_{t_s^i}^{t_s^{i+1}} \frac{P(t)}{\alpha(t)} \mathrm{d}t \,. \tag{24.19}$$

This integral equation is easily solved as $P(t)$ is a piecewise linear function and $\alpha(t)$ is a piecewise constant function:

$$\alpha(t) = a\left(t_a^i\right) \quad \text{for } t_a^i \le t < t_a^{i+1} \,. \tag{24.20}$$

Figure 24.6 shows an example of a mapping between the analysis and synthesis time instants for pitch modification by 1.5.

24.4.3 Joint Pitch and Time-Scale Modification

Based on the procedures presented above, joint pitch and time-scale modifications can easily be obtained. Given a pitch and time-scale modification factor at each analysis time instant and combining (24.17) and (24.19), the synthesis time instants can be obtained by solving the following integral equation

$$t_s^{i+1} - t_s^i = \frac{1}{t_v^{i+1} - t_v^i} \int_{t_v^i}^{t_v^{i+1}} \frac{P(t)}{\alpha(t)} \mathrm{d}t \,. \tag{24.21}$$

The synthesis time instants determined by the procedures above are not, in general, univocally associated with the analysis time instants (see for example Figs. 24.5 and 24.6). A solution consists of replacing the virtual pitch marks by the nearest analysis time instant. In this case, however, another problem arises: two or more successive frames could be the same and then the harmonic parameters (amplitudes and phases) will not vary within the frame, meaning that a high-quality modified synthetic signal is not produced. It follows therefore that the repeated analysis time instants should be elimi-

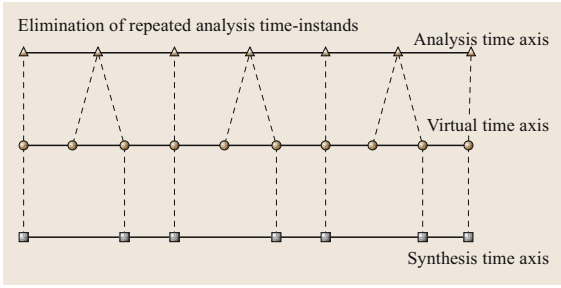


Fig. 24.7 Elimination of repeated analysis time instants from the example in Fig. 24.6

nated. As an example, in Fig. 24.7 the repeated analysis time instants in Fig. 24.6 are eliminated.

24.4.4 Energy Modification

Energy modification is performed by associating an intensity scale factor, $c(t_s^i)$, at each synthesis time instant. Then, the harmonic amplitudes are multiplied by the square root of the current harmonic intensity scale factor

$$A'_k(t_s^i) = \sqrt{c(t_s^i)}\,A_k(t_s^i) \quad \text{for } k = 1, \ldots, K\,, \tag{24.22}$$

where K is the number of harmonics. The modified amplitudes should then be used in estimating the amplitude envelope (Sect. 24.2.2).

24.4.5 Generating the Source Modified Speech Signal

In synthesis, the first step is the computation of the harmonic amplitudes and phases at the shifted harmonic frequencies. Note that in the case of time-scale modifications, the original amplitudes and phase may be preserved. In general, therefore, the amplitudes and phases are obtained by sampling the phase and amplitude envelopes at the corresponding harmonic frequencies. In the case of pitch modification, the phase and amplitude envelope should be sampled using the modified fundamental frequency: $f'_0(t_a^i) = \alpha(t_a^i) f_0(t_a^i)$. Given a spectrum, this results in a different number of harmonics from those initially included in the spectrum. When $\alpha(t_a^i) > 1$, fewer harmonics are included in the spectrum, and when $\alpha(t_a^i) < 1$, more harmonics are included. This means that the initial energy of the signal will be changed. Therefore, the amplitudes of the shifted harmonics are normalized in such a way that the final energy of the pitch-modified signal is equal to the energy of the unmodified one.

Using the new synthesis time instants and the modified set of parameters (amplitudes and phases) corresponding to each time instant, the source modified speech signal is obtained in exactly the same way as shown in Sect. 24.2.3. Examples of time-scale modification and pitch modification, both using a modification factor of 1.3, are depicted in Figs. 24.8 and 24.9, respectively.

An alternative to the parametric harmonic model for source modifications is the time-domain pitch synchronous overlap add (TD-PSOLA) [24.16]. TD-PSOLA relies heavily on the source–filter speech production model, although the parameters of this model are not estimated explicitly. TD-PSOLA is characterized by simplicity and low computational complexity, allowing good-quality prosodic modifications of speech. It is widely adopted for text-to-speech synthesis based on concatenation of acoustic units like diphones. Other methods similar to TD-PSOLA have also been proposed, including multiband resynthesis

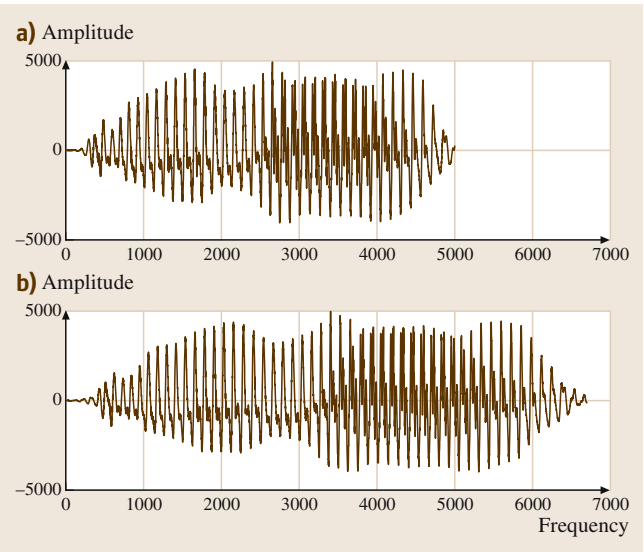


Fig. 24.8 **(a)** Original speech signal. **(b)** Time-scaled by 1.3. Time is provided in samples (sampling frequency: 16 kHz)

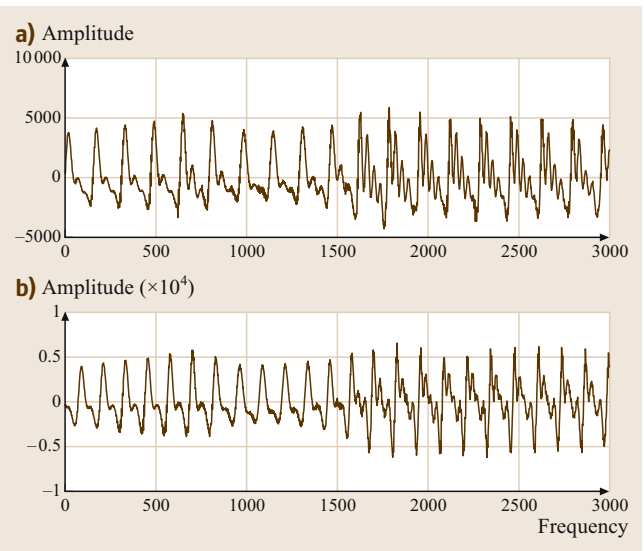


Fig. 24.9 **(a)** Original speech signal. **(b)** Pitch modified by 1.3. Time is provided in samples (sampling frequency: 16 kHz)

overlap add (MBROLA) [24.1]. The synchronized overlap add (SOLA) [24.17] and the waveform similarity overlap add (WSOLA) [24.18] methods have mainly been proposed for time-scale modification. In SOLA and WSOLA, successive speech frames to be overlapped are cross-correlated, providing the time shift to ensure that the two overlapping frames are synchronized and thus add coherently. TD-PSOLA relies on GCIs (pitch marks for voice sounds) to synchronize the speech frames. The use of pitch marks allows TD-PSOLA to apply a simple mechanism for pitch modification; this is not possible for the SOLA and WSOLA techniques. Examples of how to apply TD-PSOLA for speech modifications using the mapping of analysis and synthesis time instants are provided in Chap. 19.

Nonparametric approaches such as TD-PSOLA, SOLA, and WSOLA, do not allow complex modifications of the signal, such as increasing the degree of friction, or changing the amplitude and phase relationships between the pitch harmonics. Another major drawback is the manipulation of *noise-like* sounds presented in speech. For example, TD-PSOLA eliminates/duplicates short-time waveforms extracted from the original speech signal by windowing. When this approach is applied to unvoiced fricatives a tonal noise is produced because the repetition of segments of a *noise-like* signal produces an artificial long-time autocorrelation in the output signal, perceived as some sort of periodicity [24.15]. A simple solution to this problem consists of reversing the time-axis whenever the TD-PSOLA algorithm needs to duplicate *unvoiced* short-time signals [24.15]. This solution *reduces* the undesirable correlation in the output signal but the tonal quality does not completely disappear. This solution cannot be applied when the time-scale factor is greater than 2. Moreover, this solution cannot be used when voiced fricative frames are processed.

On the other hand, sinusoidal models have been found to be an efficient representation of voiced speech [24.5]. For a flexible representation of the unvoiced sounds and for high-quality pitch and time-scale modification of speech, hybrid [i. e., harmonic plus noise (HNM) [24.6]] models are more suitable.

24.5 Filter Modifications

Next, we will consider the case of filter modification *with* a specific target. This provides a more-general framework for filter modification than without a specific target, since it has a higher time resolution; the modification filter should be changed faster than in the case where a nonspecific target is provided. In this context, a set of source and target spectral envelopes is assumed given that an appropriate representation of the vocal tract spectral envelope is provided (i. e., using cepstral coefficients, line spectrum frequencies, mel frequency cepstral coefficients). To convert the source spectral envelope to the target spectral envelope, a training (or learning step) is necessary. During this step, a conversion function is trained. For this purpose, the source and the target speaker utter the same sentences.

One of the earliest approaches to the filter conversion is the mapping codebook method [vector quantization (VQ)] of *Abe* et al. [24.19], which was originally introduced for speaker adaptation by *Shikano* et al. [24.20]. The basic idea of this technique is to make mapping codebooks that represent the correspondence between the two speakers. A conversion of acoustic features from one speaker to another is therefore reduced to the problem of mapping the codebooks of the two speakers [24.19]. The main shortcoming of this method is the fact that the acoustic space of the converted signal is limited to a discrete set of envelopes. To avoid the limitations of the discrete space represented by VQ, a fuzzy vector quantization (FVQ) has been proposed by *Kuwabara* et al. [24.21]. A quite different approach, also based on VQ, has been proposed by *Iwahashi* et al. [24.22] using speaker interpolation. The use of linear multivariate regression (LMR) for *mapping* one class from the VQ space of the source speaker to the *corresponding* class in the VQ space of the target speaker has been proposed by *Valbret* et al. [24.23]. In the same communication [24.23], *Valbret* et al. proposed a spectral transformation approach based on dynamic frequency warping (DFW). In LMR a simple linear transformation function for each class has been proposed, while in DFW a third-order polynomial is used. All these methods have been developed in the context of VQ. Most authors agree that the mapping codebook approach, although it provides an impressive perceptive voice conversion effect, is plagued by poor quality and lack of robustness [24.24]. Approaches based on LMR and DFW also introduce discontinuities into the spectral information, as the acoustic space of a speaker

is partitioned in discrete regions. DFW succeeds in moving the formant frequencies but it has little or no effect on their amplitudes and bandwidths. Mapping functions have also been proposed using more-robust modeling compared to VQ of the acoustic space of a speaker based on the Gaussian mixture model (GMM). Assuming that the source and target vectors obtained from the speakers acoustic space are jointly Gaussian, a continuous probabilistic mapping function based on GMM has been proposed [24.25, 26]. A similar mapping function has been proposed by *Kain* et al. [24.27], jointly modeling the source and target vectors with GMM.

All of these techniques are based on parallel training data, where both the source and target speaker utter the same sentence. Then, a DTW is used to align the two signals in time in order to extract the aligned source and target training vectors. Approaches without the requirement of parallel data have also been proposed in the literature [24.28, 29]. However, using the same mapping functions for parallel and nonparallel data, it has been shown that training with parallel data provides better conversion results [24.28].

In the following, we will present a state-of-the-art system for filter modification (spectral conversion) based on GMM making use of parallel data [24.26].

24.5.1 The Gaussian Mixture Model

The Gaussian mixture density is a weighted sum of m component densities and given by the equation

$$p(\boldsymbol{x}|\Theta) = \sum_{i=1}^{m} \alpha_i \, p_i(\boldsymbol{x}|\theta_i) \,, \tag{24.23}$$

where $\boldsymbol{x} = [x_1 \; x_2 \; x_3 \; \cdots \; x_p]^{\mathrm{T}}$ is a p-by-1-dimensional random vector, $p_i(\boldsymbol{x}|\theta_i)$, for $i = 1, \cdots, m$, are the component densities and α_i are the mixture weights. Each component density, $p_i(\boldsymbol{x}|\theta_i)$, is a p-dimensional normal distribution

$$p_i(\boldsymbol{x}|\theta_i) = N(\boldsymbol{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \tag{24.24}$$

with $\boldsymbol{\mu}_i$ the p-by-1 mean vector and $\boldsymbol{\Sigma}_i$ the p-by-p covariance matrix. The mixture weights, α_i, are normalized positive scalar weights ($\sum_{i=1}^{m} \alpha_i = 1$ and $\alpha_i \geq 0$). This ensures that the mixture is a true probability density function (PDF). The complete Gaussian mixture density is parameterized by the mixture weights, the mean vectors and the covariance matrices from all component densities, which is represented by the notation,

$$\Theta = (\alpha_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \,, \quad i = 1, \cdots, m \,. \tag{24.25}$$

The Gaussian mixture model (GMM) is a classic parametric model used in many pattern-recognition techniques [24.30] and speech applications such as speaker recognition [24.31]. In the GMM context, a speaker's voice is characterized by m acoustic classes representing some broad phonetic events, such as vowels, nasal or fricatives. The probabilistic modeling of an acoustic class is important since there is variability in features coming from the same class due to variations in pronunciation and co-articulation. Thus, the mean vector $\boldsymbol{\mu}_i$ represents the average features for the acoustic class ω_i, and the covariance matrix $\boldsymbol{\Sigma}_i$ models the variability of features within the acoustic class.

GMM parameters are usually estimated by a standard iterative parameter estimation procedure, which is a special case of the *expectation-maximization* (EM) algorithm [24.32, 33]. Initialization of the algorithm may be provided by VQ.

24.6 Conversion Functions

Let $\boldsymbol{x}_t$ and $\boldsymbol{y}_t$ be a set of p-dimensional vectors corresponding to the spectral envelopes of the source and the target speaker, respectively, where $t = 1, \ldots, n$. It is therefore assumed that these vectors have been obtained by speech samples from both speakers that have been aligned in time using a classic dynamic time-alignment algorithm [24.34]. We also assume that a Gaussian mixture model ($\alpha_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i$, for $i = 1, \ldots, m$) has been fitted to the source vectors ($\boldsymbol{x}_t, t = 1, \ldots, n$).

It is worth noting that, if we take the limit case where the GMM is reduced to a single class and if the source vectors $\boldsymbol{x}_t$ follow a Gaussian distribution $N(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ and that the source and target vectors are jointly Gaussian, the minimum mean-square error estimate of the target vector is given by [24.35]

$$E[\boldsymbol{y}|\boldsymbol{x} = \boldsymbol{x}_t] = \boldsymbol{\nu} + \boldsymbol{\Gamma}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_t - \boldsymbol{\mu}) \,, \tag{24.26}$$

where $E[]$ denotes expectation, and $\boldsymbol{\nu}$ and $\boldsymbol{\Gamma}$ are, respectively, the mean target vector

$$\boldsymbol{\nu} = E[\boldsymbol{y}] \,,$$

and the cross-covariance matrix of the source and target vectors

$$\boldsymbol{\Gamma} = E[(\boldsymbol{y} - \boldsymbol{v})(\boldsymbol{x} - \boldsymbol{\mu})^{\mathrm{T}}] ,$$

where the superscript T denotes transposition [24.36].

In [24.27], a direct extension of (24.26) to the GMM case was proposed. However, such an extension is not supported by the theory of statistics. Moreover, such an extension makes the assumption of a one-to-one correspondence between the source and target space, which is not valid in practice. To overcome these difficulties, and motivated by (24.26) the following conversion function between the source and the target data has been proposed [24.25]:

$$\mathcal{F}(\boldsymbol{x}_t) = \sum_{i=1}^{m} P(\omega_i|\boldsymbol{x}_t)\left[\boldsymbol{v}_i + \boldsymbol{\Gamma}_i \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}_t - \boldsymbol{\mu}_i)\right] . \tag{24.27}$$

The conversion function $\mathcal{F}$ is entirely defined by the p-dimensional vectors $\boldsymbol{v}_i$ and the p-by-p matrices $\boldsymbol{\Gamma}_i$, for $i = 1, \ldots, m$ (where m is the number of mixture components). This means that $\boldsymbol{v}_i$ and $\boldsymbol{\Gamma}_i$ are the parameters to be estimated. The parameters of the conversion function are computed by least squares optimization on the learning data so as to minimize the total squared conversion error

$$\epsilon = \sum_{t=1}^{n} ||\boldsymbol{y}_t - \mathcal{F}(\boldsymbol{x}_t)||^2 . \tag{24.28}$$

Since the conversion function given by (24.27) is linear, the optimization of its parameters is equivalent to the resolution of a set of linear equations in the least-squares sense. Details of the minimization of (24.28) may be found in [24.37]. The mapping function (24.27) can be used with full or diagonal covariance matrices. Note that the conversion function is reduced to

$$\mathcal{F}(\boldsymbol{x}_t) = \sum_{i=1}^{m} P(\omega_i|\boldsymbol{x}_t)\boldsymbol{v}_i \tag{24.29}$$

if the correction term that depends on the difference between the source vector $\boldsymbol{x}_t$ and the mean of the GMM component $\boldsymbol{\mu}_i$ in (24.27) is omitted. This reduced conversion function is similar to the formula proposed by *Abe* et al. [24.19] in the mapping codebook approach. Comparing (24.29) and (24.27) it follows that in VQ-type mapping functions the variability of the transformed spectral envelope is strongly restricted.

An example of filter conversion based on the approach described in this section has already been presented in Fig. 24.4.

24.7 Voice Conversion

Combining source and filter modifications a system that controls speaker's quality and individuality can be obtained. Continuing with the harmonic model as an example of representing speech, a speech signal produced by the source speaker is analyzed in a pitch-synchronous way (Sect. 24.2.1) to extract a set of source spectral envelopes. Given the spectral conversion function in (24.27), the target spectral envelopes in each frame are estimated by applying (24.27) to the source spectral envelopes. The next step is to apply the appropriate prosodic or source modifications in order to capture the *prosodic patterns* of the target speaker (Sect. 24.4). For this, prosodic profile analysis (i. e., characteristic articulation rate, stress, and emotions, pitch fluctuations) is required for both speakers. Determining such a profile is not a trivial task and has not yet been achieved in a convincing way. Thus, most voice conversion systems today make use

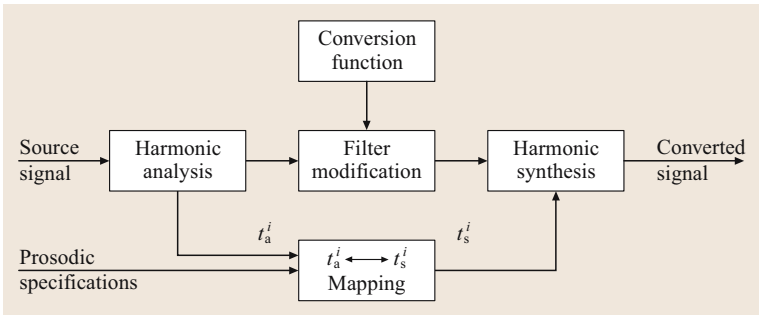


Fig. 24.10 Block diagram of a voice conversion system; t_a^i and t_s^i represent the analysis and synthesis time instants, respectively

of average prosodic modification factors. A block diagram for the conversion system based on the harmonic model and the voice conversion function is presented in Fig. 24.10.

24.8 Quality Issues in Voice Transformations

Voice transformations are usually evaluated in subjective tests. The overall impression from the results obtained from these tests is that time-scale modification is quite successful for moderate scale factors, while pitch-modified signals by pitch scale factors over 1.5 and below 0.7, suffer from various artifacts, making listeners classify the modifications as not natural. Voice conversion reaches a high score for transforming the identity of the source speaker to that of the target speaker. However, there are serious quality problems, which are mostly referred to as *muffling effects*. To improve the quality of the speech produced by various proposed voice transformation algorithms a better understanding of speech production and perception mechanisms is necessary. For example, when we want to increase the loudness of our voice while sitting in a cafeteria, we add stress to a *part* of our speech signal, like consonants, and *not* to all the speech events we produce. One hypothesis for this is that consonants carry more of the information load, which is connected with the intelligibility of the message we would like to transmit. According to this hypothesis we only increase the stress to these sounds by an amount that is sufficient to mask the cafeteria noise. Increasing the stress does not mean that the amplitudes of all the frequencies for this sound are increased. Stress means an increase of the subglottal pressure, which will result in an abrupt glottal closure by accentuating the Bernoulli effect on airflow through the glottis [24.38]. This corresponds to more energy *mostly* at high frequencies. From this example, it is obvious that even a simple intensity modification is not as simple as we thought. Continuing the above example, the increase of the subglottal pressure will increase the tension in the vocal folds, resulting in an increase of the pitch. This shows that modifying one parameter may require the modification of another as well.

In most Western languages consonants (we recall that consonants carry important information load) are shorter in duration than vowels (which carry more-prosodic information). Our perceptual system requires some time to process the perceived sounds. When we want to speak faster, we somehow *protect* the consonants. *Pickett* [24.39] has done extensive studies on the degree of change in vowels and consonants in speaking at a faster or slower rate. In [24.39], it was reported that, when going from normal to the faster rate, the vowels were compressed by 50% while the consonants were compressed by 26%. However, going from the slowest to the fastest rate, both vowels and consonants were compressed by about 33% [24.38]. This shows that time-scale modifications should take into account phonetic information. Speaking at faster or slower rate again introduces modifications in pitch values since there are fluctuations in the subglottal pressure. This means that time-scale modifications should be performed jointly with pitch modifications.

In the source–filter theory presented at the beginning of the chapter, it was assumed that glottal airflow source is not influenced by the vocal tract. In reality, there is a nonlinear coupling between the source and the filter. Results from studies on the fine structure of the glottal airflow derivative waveform show that an increase in the first-formant bandwidth and modulation of the first-formant frequency occurs during the glottal open phase [24.40]. Obviously, when pitch modification is applied, these interactions should be respected.

Attempts have been made to incorporate some of these observations into the modification algorithms. In [24.41], a higher intelligibility score was achieved for time-scale-modified speech signals when nonstationarity measurements in the signal were taken into account. In [24.42], the interaction between pitch and spectral envelopes was modeled in a statistical way. This was used to postprocess pitch-modified signals. Perceptual tests have shown that this postprocessing improved the naturalness of the pitch-modified signal.

To improve further the quality of voice transformations more effort should be made taking into account nonlinear phenomena during the production process and results from the natural language processing area. In other words, voice transformation requires more than just modeling of the speech signal; it requires *understanding* of the speech process (production, perception, and language).

24.9 Summary

In this chapter, we have described voice transformations through a simple harmonic representation of speech. We began with the description of the basic source–filter theory for the production of speech and providing a mathematical description of speech production using this theory in the context of a harmonic model. We used this description to define voice transformation by specifying modifications for the source, for the filter, and their combination. We then provided formal definitions of these modifications and their application in the context of the harmonic model was also derived and a set of conditions for the pitch synchronous analysis of speech was described. Techniques for filter modifications were discussed and a state-of-the-art method based on a GMM description of the acoustic space of a speaker was developed. A mapping function that made use of the complete description of each component of the GMM was provided. Finally, we discussed speech quality issues related to voice transformations and noted that, for improving speech quality in the future, we need to give more realism to the source–filter model by taking into account the nonlinear coupling between the source and filter and to processes related to our perception system. In other words, just modeling the speech *signal* may be enough for transmission through the networks, but it is not sufficient for modifying the signal in a way that is perceived by humans to be natural. For this, we need to *understand* speech and then *develop* algorithms able to incorporate this understanding.

References

24.1 T. Dutoit: *An Introduction to Text-to-Speech Synthesis* (Kluwer Academic, Dordrecht 1997)

24.2 J.D. Markel, A.M. Gray: *Linear Prediction of Speech* (Springer, Berlin, Heidelberg 1976)

24.3 B. Atal, J. Remde: A new model of LPC excitation for producing natural-sounding speech at low bit rates, Proc. IEEE ICASSP, Vol. 7 (1982) pp. 614–617

24.4 M.R. Schroeder, B.S. Atal: Code-excited linear prediction (CELP): High-quality speech at very low bit rates, Proc. IEEE ICASSP, Vol. 10 (1985) pp. 937–940

24.5 R.J. McAulay, T.F. Quatieri: Speech analysis/synthesis based on a sinusoidal representation, IEEE ICASSP **34**, 744–754 (1986)

24.6 Y. Stylianou: Modeling speech based on harmonic plus noise models. In: *Nonlinear Speech Modeling and Applications*, ed. by G. Chellot, A. Esposito, M. Faundez (Springer, Berlin, Heidelberg 2005) pp. 375–383

24.7 A.V. Oppenheim, R.W. Schafer: Homomorphic analysis of speech, IEEE Trans. Audio Electroacoust. **16**, 221–228 (1968)

24.8 D.B. Paul: The spectral envelope estimation vocoder, IEEE ICASSP **29**, 786–794 (1981)

24.9 O. Cappé, E. Moulines: Regularization techniques for discrete cepstrum estimation, IEEE Signal Process. Lett. **3**(4), 100–102 (1996)

24.10 A.V. Oppenheim, R.W. Schafer: *Discrete-Time Signal Processing* (Prentice Hall, Englewood Cliffs 1989)

24.11 Y. Stylianou, J. Laroche, E. Moulines: High-quality speech modification based on a harmonic + noise model, Proc. Eurospeech, Vol. 95 (1995) pp. 451–454

24.12 T.F. Quatieri, R.J. McAulay: Shape invariant time-scale and pitch modification of speech, IEEE ICASSP **40**, 497–510 (1992)

24.13 R.J. McAulay, T.F. Quatieri: Low-rate speech coding based on the sinusoidal model. In: *Advances in Speech Signal Processing*, ed. by S. Furui, M. Sondhi (Marcel Dekker, New York 1991) pp. 165–208, Chap. 6

24.14 L. Almeida, F. Silva: Variable-frequency synthesis: An improved harmonic coding scheme., Proc. IEEE ICASSP, Vol. 9 (1984) pp. 437–440

24.15 E. Moulines, J. Laroche: Techniques for pitch-scale and time-scale transformation of speech. Part I. Non parametric methods, Speech Commun. **16**, 175–205 (1995)

24.16 E. Moulines, F. Charpentier: Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones, Speech Commun. **9**, 453–467 (1990)

24.17 S. Roucos, A. Wilgus: High-quality time-scale modification of speech, Proc. IEEE ICASSP (1985) pp. 493–496

24.18 W. Verhelst, M. Roelands: An overlap-add technique based on waveform similarity (wsola) for high quality time-scale modification of speech, Proc. IEEE ICASSP, Vol. 2 (1993) pp. 554–557

24.19 M. Abe, S. Nakamura, K. Shikano, H. Kuwabara: Voice conversion through vector quantization, Proc. IEEE ICASSP, Vol. 1 (1988) pp. 655–658

24.20 K. Shikano, K. Lee, R. Reddy: Speaker adaptation through vector quantization, Proc. IEEE ICASSP, Vol. 11 (1986) pp. 2643–2646

24.21 H. Kuwabara, Y. Sagisaka: Acoustic characteristics of speaker individuality: Control and conversion, Speech Commun. **16**(2), 165–173 (1995)

24.22 N. Iwahashi, Y. Sagisaka: Speech spectrum transformation based on speaker interpolation, Proc. IEEE ICASSP, Vol. 1 (1994) pp. 461–464

24.23 H. Valbret, E. Moulines, J. Tubach: Voice transformation using PSOLA techinques, Speech Commun. **11**(2-3), 175–187 (1992)

24.24 H. Mizuno, M. Abe: Voice conversion algorithm based on piecewise linear conversion rule of formant frequency and spectrum tilt, Speech Commun. **16**, 153–164 (1995)

24.25 Y. Stylianou, O. Cappé, E. Moulines: Statistical methods for voice quality transformation, Proc. Eurospeech, Vol. 95 (1995) pp. 447–450

24.26 Y. Stylianou, O. Cappé, E. Moulines: Continuous probabilistic transform for voice conversion, IEEE Trans. Speech Audio Process. **6**(2), 131–142 (1998)

24.27 A. Kain, M. Macon: Spectral voice conversion for text-to-speech synthesis, Proc. IEEE ICASSP, Vol. 5 (1998) pp. 285–288

24.28 A. Mouchtaris, J.V. derSpiegel, P. Mueller: Non parallel training for voice conversion based on a parameter adaptation, IEEE Trans. Audio Speech Language Process. **14**(3), 952–963 (2006)

24.29 D. Suendermann, H. Hoege, A. Bonafonte, H. Ney, A. Black, S. Narayanan: Text-independent voice conversion based on unit selection, Proc. IEEE ICASSP, Vol. 1 (2006) pp. 81–84

24.30 R.O. Duda, P.E. Hart: *Pattern Classification and Scene Analysis* (Wiley, New York 1973)

24.31 R.C. Rose, D.A. Reynolds: Text independent speaker identification using automatic acoustic segmentation, Proc. IEEE ICASSP, Vol. 1 (1990) pp. 293–296

24.32 A.P. Dempster, N.M. Laird, D.B. Rubin: Maximum likelihood from incomplete data via the EM algorithm (methodological), J. R. Stat. Soc. B **39**(1), 1–22 (1977)

24.33 A.P. Dempster, N.M. Laird, D.B. Rubin: Maximum likelihood from incomplete data via the EM algorithm (discussion), J. R. Stat. Soc. B **39**(1), 22–38 (1977)

24.34 L.R. Rabiner, B.-H. Juang: *Fundamentals of Speech Recognition* (Prentice Hall, Upper Saddle River 1993)

24.35 S.M. Kay: *Fundamentals of Statistical Signal Processing: Estimation Theory*, PH Signal Process. Ser. (Prentice Hall, Upper Saddle River 1993)

24.36 C. Chatfield, A.J. Collins: *Introduction to Multivariate Analysis* (Chapman Hall, Boca Raton 1980)

24.37 Y. Stylianou: *Harmonic plus Noise Models for Speech, Combined with Statistical Methods, for Speech and Speaker Modification*, Ph.D. Thesis (Ecole Nationale Supèrieure des Télécommunications, Paris 1996)

24.38 T.F. Quatieri: *Discrete-Time Speech Signal Processing* (Prentice Hall, Englewood Cliffs 2002)

24.39 J.M. Pickett: *The Sounds of Speech Communication* (Pro-Ed, Austin 1980)

24.40 C. Jankowski: *Fine Structure Features for Speaker Identification*, Ph.D. Thesis (Massachusetts Institute of Technology, Cambridge 1996)

24.41 D. Kapilow, Y. Stylianou, J. Schroeter: Detection of non-stationarity in speech signals and its application to time-scaling, Proc. Eurospeech, Vol. 99 (1999) pp. 2307–2310

24.42 A. Kain, Y. Stylianou: Stochastic modeling of spectral adjustment for high quality pitch modification, Proc. IEEE ICASSP, Vol. 2 (2000) pp. 949–952

29. A Machine Learning Framework for Spoken-Dialog Classification

C. Cortes, P. Haffner, M. Mohri

One of the key tasks in the design of large-scale dialog systems is classification. This consists of assigning, out of a finite set, a specific category to each spoken utterance, based on the output of a speech recognizer. Classification in general is a standard machine-learning problem, but the objects to classify in this particular case are word lattices, or weighted automata, and not the fixed-size vectors for which learning algorithms were originally designed. This chapter presents a general kernel-based learning framework for the design of classification algorithms for weighted automata. It introduces a family of kernels, *rational kernels*, that combined with support vector machines form powerful techniques for spoken-dialog classification and other classification tasks in text and speech processing. It describes efficient algorithms for their computation and reports the results of their use in several difficult spoken-dialog classification tasks based on deployed systems. Our results show that rational kernels are easy to design and implement, and lead to substantial improvements of the classification accuracy. The chapter also provides some theoretical results helpful for the design of rational kernels.

29.1 Motivation

A critical problem for the design of large-scale spoken-dialog systems is to assign a category, out of a finite set, to each spoken utterance. These categories help guide the dialog manager in formulating a response to the speaker. The choice of categories depends on the application, they could be for example *referral* or *pre-certification* for a health-care company dialog system, or *billing services* or *credit* for an operator-service system.

To determine the category of a spoken utterance, one needs to analyze the output of a speech recognizer. Figure 29.1 is taken from a customer-care application. It illustrates the output of a state-of-the-art speech recognizer in a very simple case where the spoken utterance is "Hi, this is my number." The output is an acyclic weighted automaton called a *word lattice*. It compactly represents the recognizer's best guesses. Each path is la-

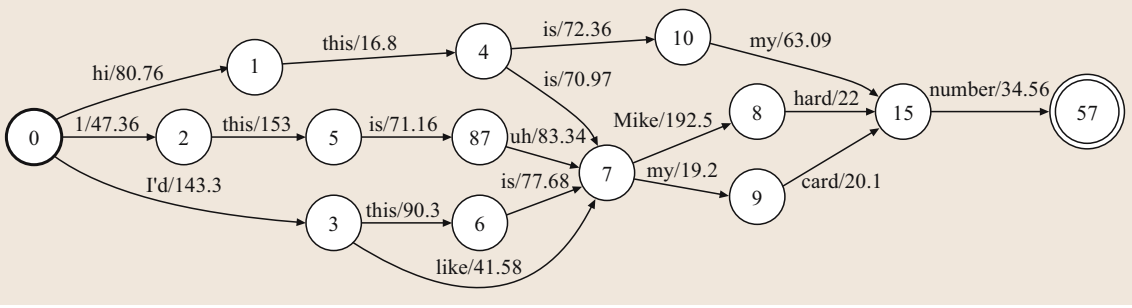


Fig. 29.1 Word lattice output of a speech recognition system for the spoken utterance "Hi, this is my number"

beled with a sequence of words and has a score obtained by summing the weights of the constituent transitions. The path with the lowest score is the recognizer's best guess, in this case: *I'd like my card number*.

This example makes evident that the error rate of conversational speech recognition systems is still too high in many tasks to rely only on the one-best output of the recognizer. Instead, one can use the full word lattice, which contains the correct transcription in most cases. This is indeed the case in Fig. 29.1, since the top path is labeled with the correct sentence. Thus, in this chapter, spoken-dialog classification is formulated as the problem of assigning a category to each word lattice.

Classification in general is a standard machine-learning problem. A classification algorithm receives a finite number of labeled examples which it uses for training, and selects a hypothesis expected to make few errors on future examples. For the design of modern spoken-dialog systems, this training sample is often available. It is the result of careful human labeling of spoken utterances with a finite number of predetermined categories of the type already mentioned.

However, most classification algorithms were originally designed to classify fixed-size vectors. The objects to analyze for spoken-dialog classification are word lattices, each a collection of a large number of sentences with some weight or probability. How can standard classification algorithms such as support vector machines [29.1] be extended to handle such objects?

This chapter presents a general framework and solution for this problem, which is based on *kernels methods* [29.2, 3]. Thus, we shall start with a brief introduction to kernel methods (Sect. 29.2). Section 29.3 will then present a kernel framework, *rational kernels*, that is appropriate for word lattices and other weighted automata. Efficient algorithms for the computation of these kernels will be described in Sect. 29.4. We also report the results of our experiments using these methods in several difficult large-vocabulary spoken-dialog classification tasks based on deployed systems in Sect. 29.5. There are several theoretical results that can guide the design of kernels for spoken-dialog classification. These results are discussed in Sect. 29.6.

29.2 Introduction to Kernel Methods

Let us start with a very simple two-group classification problem illustrated by Fig. 29.2 where one wishes to distinguish two populations, the blue and red circles. In this very simple example, one can choose a hyperplane to separate the two populations correctly, but there are infinitely many choices for the selection of that hyperplane. There is good theory though supporting the choice of the hyperplane that maximizes the *margin*, that is the distance between each population and the separating hyperplane. Indeed, let $\mathcal{F}$ denote the class of real-valued functions on the ball of radius R in $\mathbb{R}^N$:

$$\mathcal{F} = \{x \mapsto w \cdot x : \|w\| \leq 1, \|x\| \leq R\} . \tag{29.1}$$

Then, it can be shown [29.4] that there is a constant c such that, for all distributions D over X, with probability at least $1-\delta$, if a classifier $\mathrm{sgn}(f)$, with $f \in \mathcal{F}$, has margin at least ρ over m independently generated training examples, then the generalization error of $\mathrm{sgn}(f)$, or error on any future example, is no more than

$$\frac{c}{m}\left(\frac{R^2}{\rho^2}\log^2 m + \log\frac{1}{\delta}\right) . \tag{29.2}$$

This bound justifies large-margin classification algorithms such as support vector machines (SVMs). Let $w \cdot x + b = 0$ be the equation of the hyperplane, where $w \in \mathbb{R}^N$ is a vector normal to the hyperplane and $b \in \mathbb{R}$ a scalar offset. The classifier $\mathrm{sgn}(h)$ corresponding to this hyperplane is unique and can be defined with respect to the training points $x_1, \ldots, x_m$:

$$h(x) = w \cdot x + b = \sum_{i=1}^{m} \alpha_i (x_i \cdot x) + b , \tag{29.3}$$

where the α_i are real-valued coefficients. The main point we are interested in here is that, both for the construction

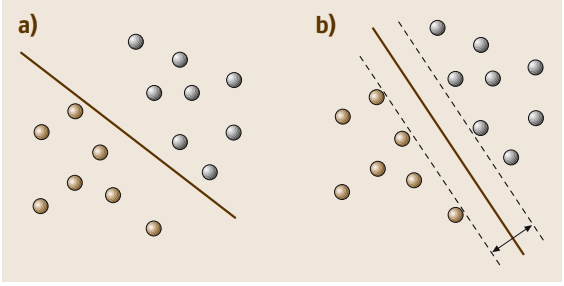


Fig. 29.2a,b Large-margin linear classification. **(a)** An arbitrary hyperplane can be chosen to separate the two groups. **(b)** The maximal-margin hyperplane provides better theoretical guarantees

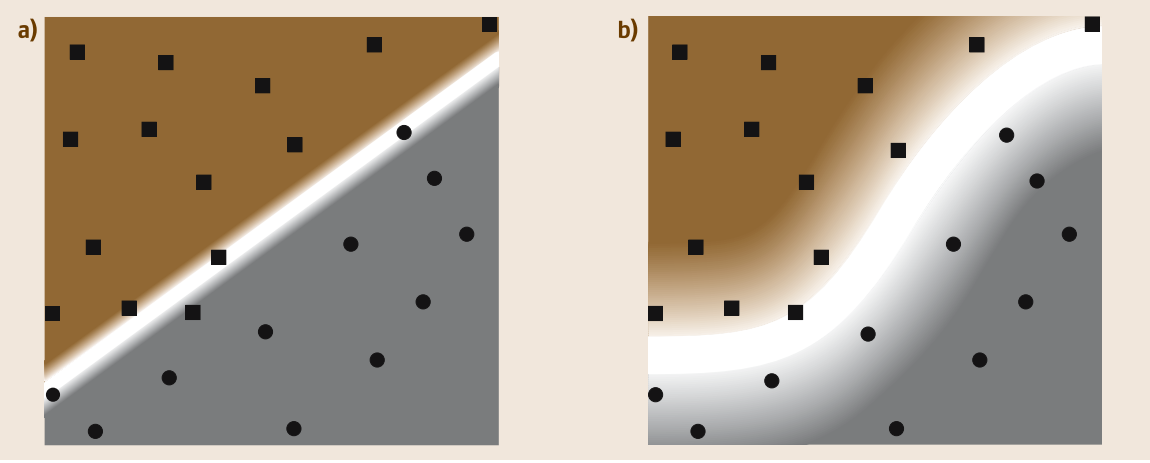


Fig. 29.3a,b Nonlinearly separable case. The classification task consists of discriminating between the solid squares and solid circles. **(a)** No hyperplane can separate the two populations. **(b)** A nonlinear mapping can be used instead

of the hypothesis and the later use of that hypothesis for classification of new examples, one needs only to compute a number of dot products between examples.

In practice, nonlinear separation of the training data is often not possible. Figure 29.3a shows an example where any hyperplane crosses both populations. However, one can use more-complex functions to separate the two sets as in Fig. 29.3b. One way to do that is to use a nonlinear mapping $\Phi : X \to F$ from the input space X to a higher-dimensional space F where linear separation is possible.

The dimension of F can truly be very large in practice. For example, in the case of document classification, one may use as features, sequences of three consecutive words (trigrams). Thus, with a vocabulary of just 100 000 words, the dimension of the feature space F is 10^{15}. On the positive side, as indicated by the error bound of (29.2), the generalization ability of large-margin classifiers such as SVMs does not depend on the dimension of the feature space but only on the margin ρ and the number of training examples m. However, taking a large number of dot products in a very high-dimensional space to define the hyperplane may be very costly.

A solution to this problem is to use the so-called *kernel trick* or *kernel methods*. The idea is to define a function $K : X \times X \to \mathbb{R}$ called a *kernel*, such that the kernel function on two examples x and y in input space, $K(x, y)$, is equal to the dot product of two examples $\Phi(x)$ and $\Phi(y)$ in feature space:

$$\forall x, y \in X, \quad K(x, y) = \Phi(x) \cdot \Phi(y) . \tag{29.4}$$

K is often viewed as a similarity measure. A crucial advantage of K is efficiency: there is no need anymore to define and explicitly compute $\Phi(x)$, $\Phi(y)$, and $\Phi(x) \cdot \Phi(y)$. Another benefit of K is flexibility: K can be arbitrarily chosen as long as the existence of Φ is guaranteed, which is called Mercer's condition. This condition is important to guarantee the convergence of training for algorithms such as SVMs. Some standard Mercer kernels over a vector space are the polynomial kernels of degree $d \in \mathbb{N}$, $K_d(x, y) = (x \cdot y + 1)^d$, and Gaussian kernels $K_\sigma(x, y) = \exp(-\|x - y\|^2/\sigma^2)$, $\sigma \in \mathbb{R}_+$.

A condition equivalent to Mercer's condition is that the kernel K be *positive definite and symmetric*, that is, in the discrete case, the matrix $[K(x_i, x_j)]_{1 \le i,j \le n}$ must be symmetric and positive semidefinite for any choice of n points $x_1, \ldots, x_n$ in X. Said differently, the matrix must be symmetric and its eigenvalues nonnegative. Thus, for the problem that we are interested in, the question is how to define positive-definite symmetric kernels for word lattices or weighted automata.

29.3 Rational Kernels

This section introduces a family of kernels for weighted automata, *rational kernels*. We will start with some preliminary definitions of automata and transducers. For the most part, we will adopt the notation

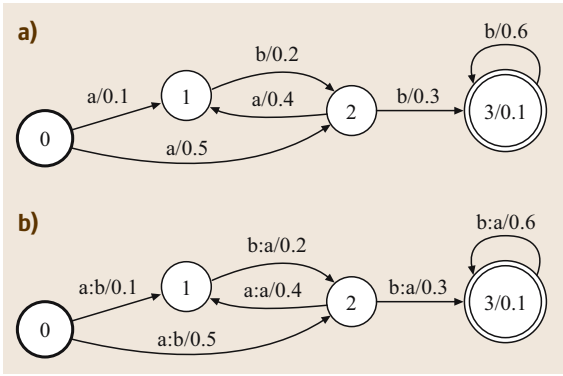


Fig. 29.4a,b Weighted automata and transducers. **(a)** Example of a weighted automaton A. $[\![A]\!](\text{abb}) = 0.1\times0.2\times0.3\times0.1+0.5\times0.3\times0.6\times0.1$. **(b)** Example of a weighted transducer T. $[\![T]\!](\text{abb}, \text{baa}) = [\![A]\!](abb)$

introduced in Chap. 28 for automata and transducers.

Figure 29.4a shows a simple weighted automaton. It is a weighted directed graph in which edges or transitions are augmented with a label and carry some weight indicated after the slash symbol. A bold circle indicates an initial state and a double circle a final state. A final state may also carry a weight indicated after the slash symbol representing the state number.

A path from an initial state to a final state is called a *successful path*. The weight of a path is obtained by multiplying the weights of constituent transitions and the final weight. The weight associated by A to a string x, denoted by $[\![A]\!](x)$, is obtained by summing the weights of all paths labeled with x. In this case, the weight associated to the string 'abb' is the sum of the weight of two paths. The weight of each path is obtained by multiplying transition weights and the final weight 0.1.

Similarly, Fig. 29.4b shows an example of a weighted transducer. Weighted transducers are similar to weighted automata but each transition is augmented with an output label in addition to the familiar input label and the weight. The output label of each transition is indicated after the colon separator. The weight associated to a pair of strings (x, y), denoted by $[\![T]\!](x, y)$, is obtained by summing the weights of all paths with input label x and output label y. Thus, for example, the weight associated by the transducer T to the pair (abb, baa) is obtained as in the automata case by summing the weights of the two paths labeled with (abb, baa).

To help us gain some intuition into the definition of a family of kernels for weighted automata, let us first consider kernels for sequences. As mentioned earlier in Sect. 29.2, a kernel can be viewed as a similarity measure. In the case of sequences, we may say that two strings are similar if they share many common substrings or subsequences. The kernel could then be for example the sum of the product of the counts of these common substrings. But how can we generalize that to weighted automata?

Similarity measures such as the one just discussed can be computed by using weighted finite-state transducers. Thus, a natural idea is to use weighted transducers to define similarity measures for sequences. We will say a that kernel K is *rational* when there exists a weighted transducer T such that

$$K(x, y) = [\![T]\!](x, y)\,, \tag{29.5}$$

for all sequences x and y. This definition generalizes naturally to the case where the objects to handle are weighted automata. We say that K is *rational* when there exists a weighted transducer T such that $K(A, B)$, the similarity measure of two automata A and B, is given by:

$$K(A, B) = \sum_{x,y} [\![A]\!](x)\cdot[\![T]\!](x, y)\cdot[\![B]\!](y)\,. \tag{29.6}$$

$T(x, y)$ is the similarity measure between two strings x and y. But, in addition, we need to take into account the weight associated by A to x and by B to y, and sum over all pairs (x, y). This definition can be generalized to the case of an arbitrary semiring where general operations other than the usual sum and multiplication are applied, which has important theoretical and algorithmic consequences and also interesting software-engineering implications [29.5].

Our definition of kernels for weighted automata is a first step towards extending SVMs to handle weighted automata. However, we also need to provide efficient algorithms for computing the kernels. The weighted automata that we are dealing with in spoken-dialog systems are word lattices that may have hundreds of thousands of states and transitions, and millions of paths, thus the efficiency of the computation is critical. In particular, the computational cost of applying existing string kernel algorithms to each pair of paths of two automata A and B is clearly prohibitive.

We also have to provide effective kernels for spoken-dialog classification and prove that they are indeed positive-definite symmetric so they can be combined with SVM for high-accuracy classification systems.

29.4 Algorithms

This section describes general, efficient algorithms for the computation of rational kernels between weighted automata and provide examples of effective positive definite symmetric kernels for spoken-dialog classification.

The outline of the algorithm is as follows. The main observation is that the sum defining rational kernels can be viewed as the sum of the weights of all the paths of a single transducer, obtained by *composing* A with T with B (see Chap. 28):

$$\sum_{x,y} [\![A]\!](x)[\![T]\!](x, y)[\![B]\!](y) = \sum_{x,y} [\![A \circ T \circ B]\!](x, y)\,. \tag{29.7}$$

The sum of the weights of all the paths of a transducer can be computed using a general single-source shortest-distance algorithm over the ordinary $(+, \times)$ semiring or the so-called forward–backward algorithm in the acyclic case [29.6]. Thus, this leads to the following general algorithm to compute $K(A, B)$ when K is a rational kernel associated to the transducer T.

- Use the composition algorithm (see Chap. 28) to compute $U = A \circ T \circ B$ in time $O(|T||A||B|)$.
- Use a general single-source shortest-distance algorithm to compute the sum of the weights of all successful paths of U [29.6]. This can be done in linear time ($O(|U|)$) when U is acyclic, which is the case when A and B are acyclic automata (word lattices).

In combination, this provides a general, efficient algorithm for computing rational kernels whose total complexity for acyclic word lattices is quadratic: $O(|T||A||B|)$. Here $|T|$ is a constant independent of the automata A and B for which $K(A, B)$ needs to be computed.

The next question that arises is how to define and compute kernels based on counts of substrings or subsequences as previously discussed in Sect. 29.3. For weighted automata, we need to generalize the notion of counts to take into account the weight of the paths in each automaton and use instead the *expected counts* of a sequence.

Let $|u|_x$ denote the number of occurrences of a substring x in u. Since a weighted automaton A defines a distribution over the set of strings u, the *expected count* of a sequence x in a weighted automaton A can be defined naturally as

$$c_A(x) = \sum_{u \in \Sigma^*} |u|_x [\![A]\!](u)\,, \tag{29.8}$$

where the sum runs over Σ^*, the set of all strings over the alphabet Σ. We mentioned earlier that weighted transducers can often be used to count the occurrences of some substrings of interest in a string. Let us assume that one could find a transducer T that could compute the expected counts of all substrings of interest appearing in a weighted automaton A. Thus, $A \circ T$ would provide the expected counts of these substrings in A. Similarly, $T^{-1} \circ B$ would provide the expected counts of these substrings in B. Recall that T^{-1} is the transducer obtained by swapping the input and output labels of each transition Chap. 28. Composition of $A \circ T$ and $T^{-1} \circ B$ matches paths labeled with the same substring in $A \circ T$ and $T^{-1} \circ B$ and multiplies their weights. Thus, by definition of composition (see Chap. 28), we could compute the sum of the expected counts in A and B of the matching substrings by computing

$$A \circ T \circ T^{-1} \circ B\,, \tag{29.9}$$

and summing the weights (expected counts) of all paths using a general single-source shortest-distance algorithm.

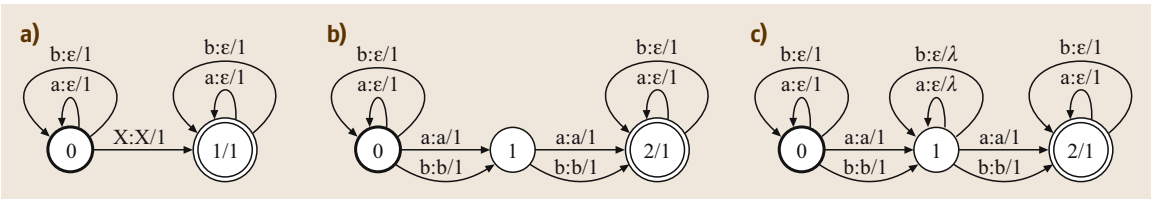


Fig. 29.5a–c Weighted transducers computing the expected counts of **(a)** all sequences accepted by the regular expression X; **(b)** all bigrams; **(c)** all gappy bigrams with penalty gap λ, over the alphabet $\{a, b\}$

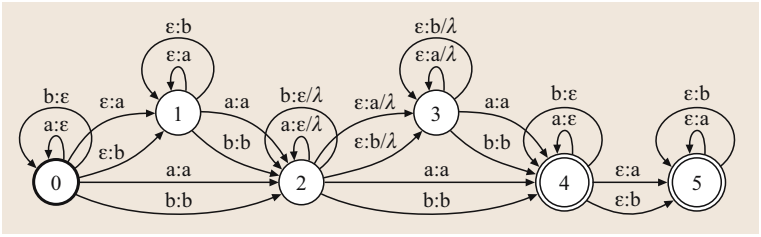


Fig. 29.6 Simple weighted transducer corresponding to the gappy bigram kernel used by [29.7] obtained by composition of the transducer of Fig. 29.5c and its inverse

Comparing this formula with (29.7) shows that the count-based similarity measure we are interested in is the rational kernel whose corresponding weighted transducer S is:

$$S = T \circ T^{-1} \,. \tag{29.10}$$

Thus, if we can determine a T with this property, this would help us naturally define the similarity measure S. In Sect. 29.6, we will prove that the kernel corresponding to S will actually be positive-definite symmetric, and can hence be used in combination with SVMs.

In the following, we will provide a weighted transducer T for computing the expected counts of substrings. It turns out that there exists a very simple transducer T that can be used for that purpose. Figure 29.5a shows a simple transducer that can be used to compute the expected counts of all sequences accepted by the regular expression X over the alphabet $\{a, b\}$. In the figure, the transition labeled with $X : X/1$ symbolizes a finite automaton representing X with identical input and output labels and all weights equal to one.

Here is how transducer T counts the occurrences of a substring x recognized by X in a sequence u. State 0 reads a prefix u_1 of u and outputs the empty string ϵ. Then, an occurrence of x is read and output identically. Then state 1 reads a suffix u_2 of u and outputs ϵ. In how many different ways can u be decomposed into $u = u_1 x u_2$? In exactly as many ways as there are occurrences of x in u. Thus, T applied to u generates exactly $|u|_x$ successful paths labeled with x. This holds for all strings x accepted by X. If T is applied to (or composed with) a weighted automaton A instead, then for any x, it generates $|u|_x$ paths for each path of A labeled with u. Furthermore, by definition of composition, each path generated is weighted with the weight of the path in A labeled with u. Thus, the sum of the weights of the paths generated that are labeled with x is exactly the expected count of x:

$$[\![A \circ T]\!](x) = c_A(x) \,. \tag{29.11}$$

Thus, there exists a simple weighted transducer T that can count, as desired, the expected counts of all strings recognized by an arbitrary regular expression X in a weighted automaton A. In particular, since the set of bigrams over an alphabet Σ is a regular language, we can construct a weighted transducer T computing the expected counts of all bigrams. Figure 29.5b shows that transducer, which has only three states regardless of the alphabet size.

In some applications, one may wish to allow for a gap between the occurrences of two symbols and view two weighted automata as similar if they share such substrings (*gappy bigrams*) with relatively large expected counts. The gap or distance between two symbols is penalized using a fixed penalty factor λ, $0 \leq \lambda < 1$. A sequence kernel based on these ideas was used successfully by [29.7] for text categorization. Interestingly, constructing a kernel based on gappy bigrams is straightforward in our framework. Figure 29.5c shows the transducer T counting expected counts of gappy bigrams from which the kernel can be efficiently constructed. The same can be done similarly for higher-order gappy n-grams or other gappy substrings.

The methods presented in this section can be used to construct efficiently and, often in a simple manner, relatively complex weighted automata kernels K based on expected counts or other ideas. A single, general algorithm can then be used as described to compute efficiently $K(A, B)$ for any two weighted automata A and B, without the need to design a new algorithm for each new kernel, as previously done in the literature. As an example, the gappy kernel used by [29.7] is the rational kernel corresponding to the six-state transducer $S = T \circ T^{-1}$ of Fig. 29.6.

29.5 Experiments

This section describes the applications of the kernel framework and techniques to spoken-dialog classification.

In most of our experiments, we used simple n-gram rational kernels. An n-gram kernel k_n for two weighted automata or lattices A and B is defined by:

$$k_n(\mathrm{A},\mathrm{B}) = \sum_{|x|=n} c_{\mathrm{A}}(x)\, c_{\mathrm{B}}(x)\,. \tag{29.12}$$

As described in Sect. 29.4, k_n is a kernel of the form $T \circ T^{-1}$ and can be computed efficiently. An n-gram rational kernel K_n is simply the sum of kernels k_m, with $1 \leq m \leq n$:

$$K_n = \sum_{m=1}^{n} k_m\,.$$

Thus, the feature space associated with K_n is the set of all m-gram sequences with $m \leq n$. As discussed in the previous section, it is straightforward, using the same algorithms and representations, to extend these kernels to kernels with gaps and to many other more-complex rational kernels more closely adapted to the applications considered.

We did a series of experiments in several large-vocabulary spoken-dialog tasks using rational kernels with a twofold objective [29.8]: to improve classification accuracy in those tasks, and to evaluate the impact on classification accuracy of the use of a word lattice rather than the one-best output of the automatic speech recognition (ASR) system.

The first task we considered was that of a deployed customer-care application (*How may I help you?*, HMIHY 0300). In this task, users interact with a spoken-dialog system via the telephone, speaking naturally, to ask about their bills, their calling plans, or other similar topics. Their responses to the open-ended prompts of the system are not constrained by the system, they may be any natural language sequence. The objective of the spoken-dialog classification is to assign one or several categories or call-types, e.g., *billing credit*, or *calling plans*, to the users' spoken utterances. The set of categories is predetermined, and in this first application there are 64 categories. The calls are classified based on the user's response to the first greeting prompt: "Hello, this is AT&T. How may I help you?"

Table 29.1 indicates the size of the HMIHY 0300 datasets we used for training and testing. The training set is relatively large with more than 35 000 utterances;

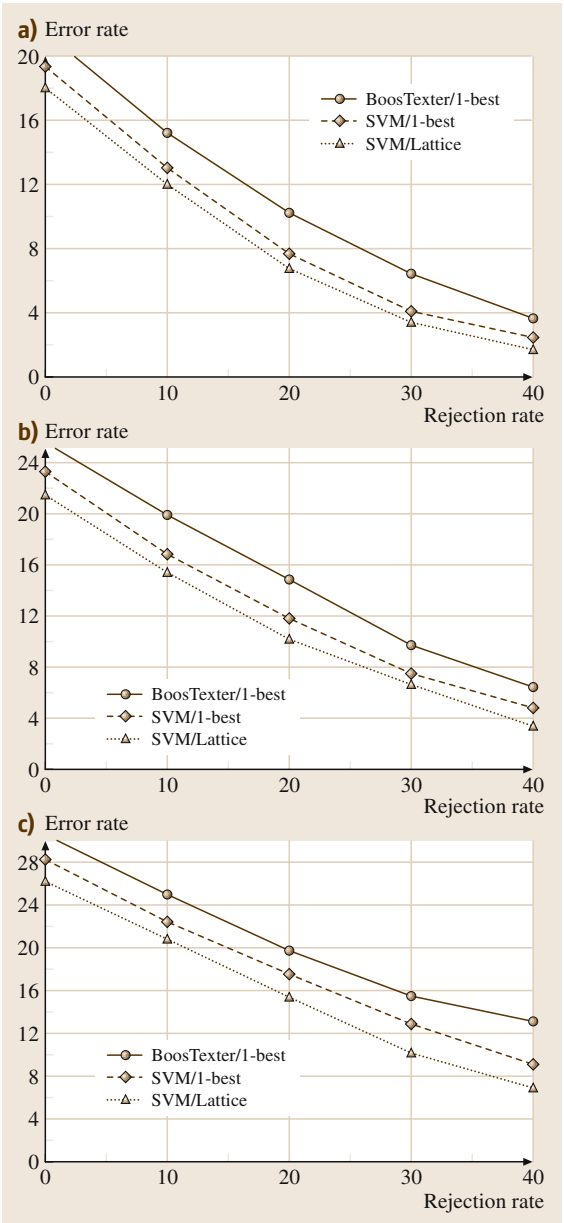


Fig. 29.7a–c Classification error rate as a function of rejection rate in **(a)** HMIHY 0300, **(b)** VoiceTone1, and **(c)** VoiceTone2

this is an extension of the one we used in our previous classification experiments with HMIHY 0300 [29.9]. In our experiments, we used the n-gram rational kernels

Table 29.1 Key characteristics of the three datasets used in the experiments. The fifth column displays the total number of unigrams, bigrams, and trigrams found in the one-best output of the ASR for the utterances of the training set, that is the number of features used by BoosTexter or SVMs used with the one-best outputs. The training and testing sizes reported in columns 3 and 4 are described in the number of utterances, or equivalently the number of word lattices

Dataset	Number of classes	Training size	Testing size	Number of n-grams	ASR word accuracy %
HMIHY 0300	64	35551	5000	24177	72.5
VoiceTone1	97	29561	5537	22007	70.5
VoiceTone2	82	9093	5172	8689	68.8

just described with $n = 3$. Thus, the feature set we used was that of all n-grams with $n \leq 3$. Table 29.1 indicates the total number of distinct features of this type found in the datasets. The word accuracy of the system based on the best hypothesis of the speech recognizer was 72.5%. This motivated our use of the word lattices, which contain the correct transcription in most cases. The average number of transitions of a word lattice in this task was about 260.

Table 29.1 reports similar information for two other datasets, VoiceTone1, and VoiceTone2. These are more recently deployed spoken-dialog systems in different areas, e.g., VoiceTone1 is a task where users interact with a system related to health-care with a larger set of categories (97). The size of the VoiceTone1 datasets we used and the word accuracy of the recognizer (70.5%) make this task otherwise similar to HMIHY 0300. The datasets provided for VoiceTone2 are significantly smaller with a higher word error rate. The word error rate is indicative of the difficulty of classification task since a higher error rate implies a more noisy input. The average number of transitions of a word lattice in VoiceTone1 was about 210 and in VoiceTone2 about 360.

Each utterance of the dataset may be labeled with several classes. The evaluation is based on the following criterion: it is considered an error if the highest scoring class given by the classifier is none of these labels.

We used the AT&T FSM library [29.10] and the GRM library [29.11] for the implementation of the n-gram rational kernels K_n used. We used these kernels with SVMs, using a general learning library for large-margin classification(LLAMA), which offers an optimized multiclass recombination of binary SVMs [29.12]. Training time took a few hours on a single processor of a 2.4 GHz Intel Pentium processor Linux cluster with 2 GB of memory and 512 KB cache.

In our experiments, we used the trigram kernel K_3 with a second-degree polynomial. Preliminary experiments showed that the top performance was reached for trigram kernels and that 4-gram kernels, K_4, did not significantly improve performance. We also found that the combination of a second-degree polynomial kernel with the trigram kernel significantly improves performance over a linear classifier, but that no further improvement could be obtained with a third-degree polynomial.

We used the same kernels in the three datasets previously described and applied them to both the speech recognizer's single best hypothesis (one-best results), and to the full word lattices output by the speech recognizer. We also ran, for the sake of comparison, the BoosTexter algorithm [29.13] on the same datasets by applying it to the one-best hypothesis. This served as a baseline for our experiments.

Figure 29.7a shows the result of our experiments in the HMIHY 0300 task. It gives classification error rate as a function of rejection rate (utterances for which the top score is lower than a given threshold are rejected) in HMIHY 0300 for: BoosTexter, SVM combined with our kernels when applied to the one-best hypothesis, and SVM combined with kernels applied to the full lattices.

SVM with trigram kernels applied to the one-best hypothesis leads to better classification than BoosTexter everywhere in the range of 0–40% rejection rate. The accuracy is about 2–3% absolute value better than that of BoosTexter in the range of interest for this task, which is roughly between 20% and 40% rejection rate. The results also show that the classification accuracy of SVMs combined with trigram kernels applied to word lattices is consistently better than that of SVMs applied to the one-best alone by about 1% absolute value.

Figures 29.7b,c show the results of our experiments in the VoiceTone1 and VoiceTone2 tasks using the same techniques and comparisons. As observed previously, in many regards, VoiceTone1 is similar to the HMIHY 0300 task, and our results for VoiceTone1 are comparable to those for HMIHY 0300. The results show that the classification accuracy of SVMs combined with trigram kernels applied to word lattices is consistently better than that of BoosTexter, by more than 4% absolute value at a rejection rate of about 20%. They also demonstrate

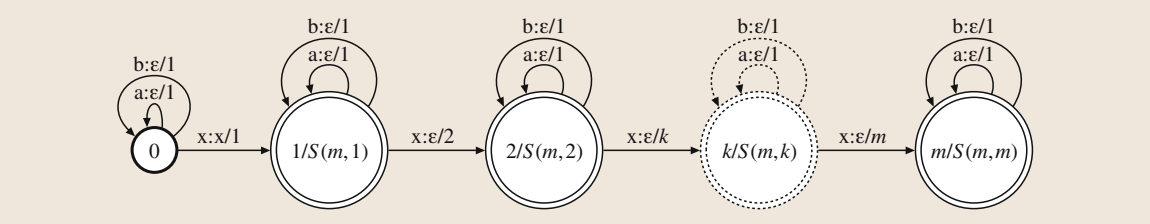


Fig. 29.8 Weighted transducer T^m for computing the m-th moment of the count of an aperiodic substring x. The final weight at state k, $k = 1, \ldots, m$, indicated after /, is $S(m, k)$, the Stirling number of the second kind, that is the number of ways of partitioning a set of m elements into k nonempty subsets, $S(m, k) = \frac{1}{k!} \sum_{i=0}^{k} \binom{k}{i} (-1)^i (k-i)^m$. The first-order transducer T_1 coincides with the transducer of Fig. 29.5a, since $S(1, 1) = 1$

more clearly the benefits of the use of the word lattices for classification in this task. This advantage is even more manifest for the VoiceTone2 task for which the speech recognition accuracy is lower. VoiceTone2 is also a harder classification task, as can be seen by comparing the plots of Fig. 29.7b. The classification accuracy of SVMs with kernels applied to lattices is more than 6% absolute value better than that of BoosTexter near the 40% rejection rate, and about 3% better than SVMs applied to the one-best hypothesis.

Thus, our experiments in spoken-dialog classification in three distinct large-vocabulary tasks demonstrated that using rational kernels with SVMs consistently leads to very competitive classifiers. They also show that their application to the full word lattices instead of the single best hypothesis output by the recognizer systematically improves classification accuracy.

We further explored the use of kernels based on other moments of the counts of substrings in sequences, generalizing n-gram kernels [29.14]. Let m be a positive integer. Let $c_{\mathrm{A}}^m(x)$ denote the m-th moment of the count of the sequence x in A defined by:

$$c_{\mathrm{A}}^m(x) = \sum_{u \in \Sigma^*} |u|_x^m \, [\![\mathrm{A}]\!](u) \,. \tag{29.13}$$

We can define a general family of kernels, denoted by K_n^m, $n, m \geq 1$ and defined by:

$$K_n^m(\mathrm{A}, \mathrm{B}) = \sum_{|x|=n} c_{\mathrm{A}}^m(x) c_{\mathrm{B}}^m(x) \,, \tag{29.14}$$

which exploit the m-th moment of the counts of substrings x in weighted automata A and B to define their similarity. *Cortes* and *Mohri* [29.14] showed that there exist weighted transducers T_n^m that can be used to compute $c_{\mathrm{A}}^m(x)$ efficiently for all n-gram sequence x and weighted automaton A. Thus, these kernels are rational kernels and their associated transducers are $T_n^m \circ T_n^{m-1}$:

$$K_n^m(\mathrm{A}, \mathrm{B}) = \sum_{x,y} [\![\mathrm{A} \circ [T_n^m \circ (T_n^m)^{-1}] \circ \mathrm{B}]\!](x, y) \,. \tag{29.15}$$

Figure 29.8 shows the weighted transducer T_1^m for aperiodic strings x, which has only $m|x|+1$ states. An aperiodic string is a string that does not admit a nonempty prefix as a suffix. The m-th moment of other strings (periodic strings) can also be computed using weighted transducers. By (29.15), the transducer T_1^m can be used to compute $K_n^m(\mathrm{A}, \mathrm{B})$ by first computing the composed transducer $A \circ [T_n^m \circ (T_n^m)^{-1}] \circ \mathrm{B}$ and then summing the weights of all the paths of this transducer using a shortest-distance algorithm [29.6] .

The application of moment kernels to the HMIHY 0300 task resulted in a further improvement of the classification accuracy. In particular, at a 15% rejection rate, the error rate was reduced by 1% absolute, which is about 6.2% relative, which is significant in this task, by using *variance kernels*, that is moment kernels of second order ($m = 2$).

29.6 Theoretical Results for Rational Kernels

In the previous sections, we introduced a number of rational kernels, e.g., kernels based on expected counts or moments of the counts and applied them to spoken-dialog tasks. Several questions arise in relation to these kernels. As pointed out earlier, to guarantee the convergence of algorithms such as SVMs, the kernels used

must be positive-definite symmetric (PDS). But, how can we construct PDS rational kernels? Are n-gram kernels and similar kernels PDS? Can we combine simpler PDS rational kernels to create more-complex ones? Is there a characterization of PDS rational kernels?

All these questions have been investigated by Cortes et al. [29.5]. The following theorem provides a general method for constructing PDS rational kernels.

Theorem 29.1
Let T be a weighted finite-state transducer over $(+, \times)$. Assume that the weighted transducer $T \circ T^{-1}$ is regulated, then $S = T \circ T^{-1}$ defines a PDS rational kernel. A weighted transducer T is said to be *regulated* when $[\![T]\!](x, y)$, the sum of the weights of the paths with input x and output y, is well defined.

Proof. We give a sketch of the proof. A full proof is given in [29.5]. Let K be the kernel associated with $S = T \circ T^{-1}$. By definition of T^{-1} and composition,

$$\forall x, y \in X, K(x, y) = \sum_z [\![T]\!](x, z)[\![T]\!](y, z) , \tag{29.16}$$

where the sum is over all strings z. Let K_n be the function defined by restricting the sum to strings of length at most k:

$$\forall x, y \in X, K_k(x, y) = \sum_{|z| \le k} [\![T]\!](x, z)[\![T]\!](y, z) . \tag{29.17}$$

Consider any ordering of all strings of length at most k: $z_1, \ldots, z_l$. For any set of n strings $x_1, \ldots, x_n$, let A be the matrix defined by $A = [[\![T]\!](x_i, z_j)]_{i \in [1,n], j \in [1,l]}$. Then, the eigenvalues of the matrix M_n defined by

$$M_n = [K_n(x_i, x_j)]_{i,j \in [1,n]} \tag{29.18}$$

are necessarily nonnegative since $M_n = AA^\top$. Thus, for any $n \le 0$, K_n is a PDS kernel. Since K is a pointwise limit of K_n, K is also PDS [29.5, 15]. □

The theorem shows that the rational kernels we considered in previous sections, e.g., count-based similarity kernels, n-gram kernels, and gappy n-gram kernels, are all PDS rational kernels, which justifies a posteriori their use in combination with SVMs. Conversely, we have conjectured elsewhere that all PDS rational kernels are rational kernels associated to transducers of the type $S = T \circ T^{-1}$ [29.5], and proved several results in support of that conjecture. In particular, for acyclic transducer, this indeed provides a characterization of PDS rational kernels.

It can also be shown that a finite sum of PDS rational kernels is a PDS rational kernel, which we used for defining n-gram kernels. More generally, the following theorem holds [29.5].

Theorem 29.2
PDS rational kernels are closed under sum, product, and Kleene closure.

Thus, one can use rational operations to create complex PDS rational kernels from simpler ones.

29.7 Conclusion

Rational kernels form an effective tool and framework for spoken-dialog classification. They are based on a general theory that guarantees in particular the positive definiteness of rational kernels based on an arbitrary $(+, \times)$-weighted transducer and thus the convergence of training for algorithms such as SVMs. General, efficient algorithms can be readily used for their computation.

Experiments in several large-vocabulary spoken-dialog tasks show that rational kernels can be combined with SVMs to form powerful classifiers and that they perform well in several difficult tasks. They also demonstrate the benefits of the use of kernels applied to word lattices.

Rational kernels form a rich family of kernels. The kernels used in the experiments we described are only special instances of this general class of kernels. Rational kernels adapted to a specific spoken-dialog task can be designed. In fact, it is often straightforward to craft prior knowledge about a task in the transducer defining these kernels. One may, for example, exclude some word sequences or regular expressions from the similarity measure defined by these kernels or emphasize the importance of others by increasing their corresponding weight in the weighted transducer.

References

29.1 C. Cortes, V.N. Vapnik: Support-vector networks, Machine Learning **20**(3), 273–297 (1995)

29.2 B.E. Boser, I. Guyon, V.N. Vapnik: A training algorithm for optimal margin classifiers. In: *Proc. 5th Annual Workshop Computational Learning Theory*, Vol. 5 (ACM, Pittsburg 1992) pp. 144–152

29.3 B. Schölkopf, A. Smola: *Learning with Kernels* (MIT Press, Cambridge 2002)

29.4 P. Bartlett, J. Shawe-Taylor: Generalization performance of support vector machines and other pattern classifiers. In: *Advances in Kernel Methods: Support Vector Learning* (MIT Press, Cambridge 1999) pp. 43–54

29.5 C. Cortes, P. Haffner, M. Mohri: Positive definite rational kernels, Proceedings of The 16th Annual Conference on Computational Learning Theory (COLT 2003) (2003), LNCS **2777**, 41–56

29.6 M. Mohri: Semiring frameworks and algorithms for shortest-distance problems, J. Automata Lang. Combinat. **7**(3), 321–350 (2002)

29.7 H. Lodhi, J. Shawe-Taylor, N. Cristianini, C. Watkins: *Text Classification Using String Kernels (NIPS 2000)* (MIT Press, Boston 2001) pp. 563–569

29.8 C. Cortes, P. Haffner, M. Mohri: Rational kernels: Theory and algorithms, J. Machine Learning Res. (JMLR) **5**, 1035–1062 (2004)

29.9 C. Cortes, P. Haffner, M. Mohri: Rational kernels. In: *Advances in Neural Information Processing Systems (NIPS 2002)*, Vol. 15 (MIT Press, Boston 2002)

29.10 M. Mohri, F.C.N. Pereira, M. Riley: The design principles of a weighted finite-state transducer library, Theoret. Comput. Sci. **231**, 17–32 (2000)

29.11 C. Allauzen, M. Mohri, B. Roark: A general weighted grammar library. In: *Proceedings of the Ninth International Conference on Automata* (CIAA, Kingston 2004)

29.12 P. Haffner, G. Tur, J. Wright: Optimizing SVMs for complex Call Classification, Proceedings ICASSP'03 (2003)

29.13 R.E. Schapire, Y. Singer: Boostexter: A boosting-based system for text categorization, Machine Learning **39**(2/3), 135–168 (2000)

29.14 C. Cortes, M. Mohri: Moment kernels for regular distributions, Machine Learning **60**(1-3), 117–134 (2005)

29.15 C. Berg, J.P.R. Christensen, P. Ressel: *Harmonic Analysis on Semigroups* (Springer, Berlin, Heidelberg 1984)

33. Environmental Robustness

J. Droppo, A. Acero

When a speech recognition system is deployed outside the laboratory setting, it needs to handle a variety of signal variabilities. These may be due to many factors, including additive noise, acoustic echo, and speaker accent. If the speech recognition accuracy does not degrade very much under these conditions, the system is called *robust*. Even though there are several reasons why real-world speech may differ from clean speech, in this chapter we focus on the influence of the *acoustical environment*, defined as the transformations that affect the speech signal from the time it leaves the mouth until it is in digital format.

Specifically, we discuss strategies for dealing with additive noise. Some of the techniques, like feature normalization, are general enough to provide robustness against several forms of signal degradation. Others, such as feature enhancement, provide superior noise robustness at the expense of being less general. A good system will implement several techniques to provide a strong defense against acoustical variabilities.

33.1 Noise Robust Speech Recognition

This chapter addresses the problem of additive noise at the input to an automatic speech recognition (ASR) system. Parts H and I in this Handbook address how to build microphone arrays for superior sound capture, or how to reduce noise for perceptual audio quality. Both of these subjects are orthogonal to the current discussion.

Microphone arrays are useful in that improved audio capture should be the first line of defense against additive noise. However, despite the best efforts of the system designer, there will always be residual additive noise. In general, speech recognition systems prefer linear array algorithms, such as beam forming, to nonlinear techniques. Although nonlinear techniques can achieve better suppression and perceptual quality, the introduced distortions tend to confuse speech recognition systems.

Furthermore, speech enhancement algorithms designed for improved human perception do not always help ASR accuracy. Most enhancement algorithms introduce some signal distortion, and the type of distortion that can be tolerated by humans and computers can be quite different.

Additive noise is common in daily life, and can be roughly categorized as either stationary or nonstationary. Stationary noise, such as that made by a computer fan or air conditioning, has a frequency spectrum that does not change over time. In contrast, the spectrum of a nonstationary noise changes over time. Some examples of nonstationary noise are a closing door, music, and other speakers' voices. In practice, no noise is perfectly stationary. Even the noises from a computer fan, an air-conditioning system, or a car will change over a long enough time period.

33.1.1 Standard Noise-Robust ASR Tasks

When building and testing noise-robust automatic speech recognition systems, there are a rich set of standards to test against.

The most popular tasks today were generated by the European Telecommunications Standards Institute's technical committee for Speech, Transmission Planning, and Quality of Service (ETSI STQ). Their AURORA digital speech recognition (DSR) working group was formed to develop and standardize algorithms for distributed and noise-robust speech recognition. As a byproduct of their work, they released a series of standard tasks [33.1] for system evaluation. Each task consists of all the necessary components for running an experiment, including data and recipes for building acoustic and language models, and scripts for running evaluations against different testing scenarios.

The Aurora 2 task is the easiest to set up and use, and is the focus of many results in this chapter. The data was derived from the TIDigits corpus [33.2], which consists of continuous English digit strings of varying lengths, spoken into a close-talking microphone. To simulate noisy telephony environments, these clean utterances were first downsampled to 8 kHz, and then additive and convolutional noise was added. The additive noise is controlled to produce noisy signals with a range of signal-to-noise ratios (SNRs) of −5–20 dB.

The noise types include both stationary and nonstationary noises, and are broken down into three sets: set A (subway, babble, car, and exhibition), set B (restaurant, street, airport, and station), and set C (subway and street). Set C contains one noise from set A, and one from set B, and also includes extra convolutional noise. There are two sets of training data, one is clean and the other is noisy. The noisy training data contains noises similar to set A. This represents the case where the system designers are able to anticipate correctly the types of noises that the system will see in practice. Test set B, on the other hand, has four different types of noises.

The acoustic model training recipe that was originally distributed with Aurora 2 was considered to be too weak. As a result, many researchers built better acoustic models to showcase their techniques. However, this made their results incomparable. To rectify this problem, a standard *complex back-end* recipe [33.3] was proposed, which is the proper model to use when performing new experiments on this task.

The Aurora 3 task is similar in complexity to the Aurora 2 task, but covers four other European languages in real car noise scenarios. Because the data is a subset of the SpeechDat car database [33.4], the noise types between Aurora 2 and Aurora 3 are quite different. The noise types chosen for Aurora 2 can be impulsive and nonstationary, but the car noise in Aurora 3 tends to be well modeled by stationary colored noise. Whereas the noisy utterances in Aurora 2 are artificially mixed, the noisy utterances of Aurora 3 were collected in actual noisy environments. Nevertheless, techniques exhibit a strong correlation in performance between Aurora 2 and Aurora 3, indicating that digitally simulated noisy speech is adequate for system evaluation.

The Aurora 4 task was developed to showcase noise-robust speech recognition for larger-vocabulary systems. Whereas the previous Aurora tasks have 10 or 11 word vocabularies, the Aurora 4 task has a 5000-word vocabulary. In much the same way that Aurora 2 was derived from the clean TIDigits corpus, the Aurora 4 task was derived from the clean Wall Street Journal (WSJ) corpus [33.5]. Noises are digitally mixed at several signal-to-noise ratios. Because of the difficulty in setting up the larger system, the Aurora 4 task is not as commonly cited in the literature. The Aurora 4 task is relevant because some techniques that work well on Aurora 2 either become intractable or fail on larger-vocabulary tasks.

Noisex-92 [33.6] is a set of data that is also useful in evaluating noise robust speech recognition systems. It consists of two CD-ROMS of audio recordings, suitable for use in artificially mixing noise with clean speech to produce noisy speech utterances. There is a great variety of noises available with the data, including voice babble, factory noise, F16 fighter jet noise, M109 tank noise, machine gun noise, and others.

Table 33.1 Word accuracy for the Aurora 2 test sets using the clean acoustic model baseline

SNR (dB)	Test set A	Test set B	Test set C	Average
Clean	99.63	99.63	99.60	99.62
20	95.02	91.71	97.02	94.58
15	85.16	78.10	92.38	85.21
10	64.50	55.75	77.78	66.01
5	34.59	29.21	51.36	38.39
0	13.61	9.75	22.82	15.40
−5	5.87	4.08	11.47	7.14
Average (0–20)	58.58	52.90	68.27	58.25

Another common evaluation task for noise robust speech recognition systems is the speech in noisy environments (SPINE) evaluation [33.7]. It was created for the Department of Defense digital voice processing consortium, to support the 2000 SPINE1 evaluation. The corpus contains 9 h 22 min of audio data, collected in simulated noisy environments where users collaborate using realistic handsets and communications channels to seek and shoot targets, similar to the game Battleship.

33.1.2 The Acoustic Mismatch Problem

To understand the extent of the problem of recognizing speech in noise, it is useful to look at a concrete example. Many of the techniques in this chapter are tested on the Aurora 2 task. Table 33.1 contains typical results from the baseline Aurora 2 system. Here, an acoustic model is trained on clean, noise-free data, and tested on data with various digitally simulated noise levels. The accuracy on clean test data averages 99.62%, which may be acceptable for some applications.

As soon as any noise is present in the test data, the system rapidly degrades. Even at a mild 20 dB signal-to-noise ratio (SNR), the system produces more than 14 times as many errors compared to clean data. (The signal-to-noise ratio is defined as the ratio of signal energy to noise energy in the received signal. It is typically measured in decibels (dB), and calculated as $10\log_{10}$[Energy(signal)/Energy(noise)]. An SNR above 30 dB sounds quite noise-free. At 0 dB SNR, the signal and noise are at the same level.) As the SNR decreases further, the problem becomes more intense. Why does an ASR system perform so poorly when presented with even mildly corrupted signals? The answer is deceptively simple. Automatic speech recognition is fundamentally a pattern matching problem. And, when a system is tested on patterns that are unlike anything used to train it, errors are likely to occur. The fundamental problem is the *acoustic mismatch* between the training and testing data.

Figure 33.1 illustrates the severity of the problem. It compares the histograms for C_1 between clean speech and moderately noisy speech. (C_1, the first cepstral coefficient, is typical speech feature used by automatic speech recognition systems.) The two histograms are quite dissimilar. Obviously, a system trained under one condition will fail under the other.

33.1.3 Reducing Acoustic Mismatch

The simplest solution to the acoustic mismatch problem is to build an acoustic model that is a better match for the test data. Techniques that can be helpful in that respect are multistyle training and model adaptation. These types of algorithms are covered in Sect. 33.2.

Another common approach to solving the acoustic mismatch problem is to transform the data so that the training and testing data tend to be more similar. These techniques concentrate on either normalizing

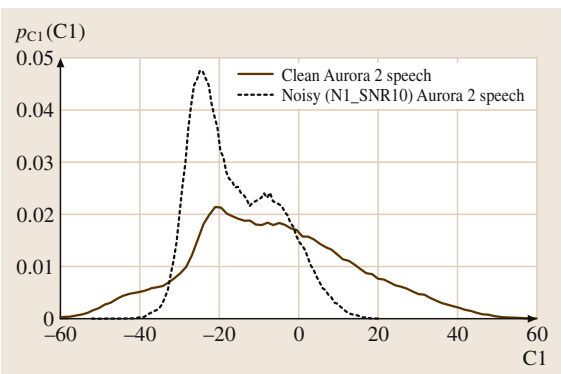


Fig. 33.1 Additive noise creates a mismatch between clean training data and noisy testing data. Here, the histogram for a clean speech feature is strikingly different from the same histogram computed from noisy speech

out the effects of noise, or learning transformations that map speech into a canonical noise-free representation. Examples of such techniques are cepstral mean normalization and stereo piecewise linear compensation for environment (SPLICE), which are covered in Sect. 33.3.

The techniques mentioned so far are powerful, but limited. They do not assume any form for the acoustic mismatch, so they can be applied to compensate for a wide range of corrupting influences. But, because they are unstructured, they need a lot of data to handle a new condition. Section 33.4 introduces a model of how noise corrupts clean speech features. Later, the model is used to derive powerful data-thrifty adaptation and normalization algorithms.

Section 33.5 presents the first class of these algorithms, which adapt the parameters of a clean acoustic model to approximate an acoustic model for noisy speech. Parallel model combination with a log-normal approximation is covered, as is vector Taylor-series (VTS) model adaptation.

The model for how additive noise corrupts clean speech features can also be used to do speech feature enhancement. Section 33.6 covers the classic technique of spectral subtraction as it can be integrated into speech recognition. It also covers vector Taylor-series speech enhancement, which has proven to be an easy and economical alternative to full model adaptation.

The last set of techniques discussed in this chapter are hybrid approaches that bridge the space between model and feature based techniques. In general, the former are more powerful, but the latter are easier to compute. Uncertainty decoding and noise adaptive training are two examples presented in Sect. 33.7, which are more powerful than a purely feature-based approach, but without the full cost of model adaptation.

33.2 Model Retraining and Adaptation

The best way to train any pattern recognition system is to train it with examples that are similar to those it will need to recognize later.

One of the worst design decisions to make is to train the acoustic model with data that is dissimilar to the expected testing input. This usually happens when the training data is collected in a quiet room using a close-talking microphone. This data will contain very good speech, with little reverberation or additive noise, but it will not look anything like what a deployed system will collect with its microphone. It's true that robustness algorithms can ameliorate this mismatch, but it is hard to cover up for a fundamental design flaw.

The methods presented in this section demonstrate that designing appropriate training data can greatly improve the accuracy of the final system.

33.2.1 Retraining on Corrupted Speech

To build an automatic speech recognition system that works in a particular noise condition, one of the best solutions is to find training data that matches this condition, train the acoustic model from this data in the normal way, and then decode the noisy speech without further processing. This is known as *matched condition* training.

If the test conditions are not known precisely, *multistyle training* [33.8] is a better choice. Instead of using a single noise condition in the training data, many different kinds of noises are used, each of which would be reasonable to expect in deployment.

Matched condition training can be simulated with sample noise waveforms from the new environments. These noise waveforms are artificially mixed with clean training data to create a synthetically noisy training data set. This method allows us to adapt the model to the new environment with a relatively small amount of data from the new environment, yet use a large amount of training data for the system.

The largest problem with multistyle training is that it makes components in the acoustic model broader and less discriminative. As a result, accuracy in any one condition is slightly worse than if matched condition training were available, but much better than if a mismatched training were used.

Tables 33.1 and 33.2 present the standard clean condition and multistyle training results from the Aurora 2 tasks. In the clean condition experiments, the acoustic model is built with uncorrupted data, even though the test data has additive noise. This is a good example of mismatched training conditions, and the resulting accuracy is quite low.

The multistyle training results in Table 33.2 are much better than the clean training results of Table 33.1. Even though the noises from set B are different from the training set, their accuracy has been improved considerably over the mismatched clean condition training. Also, notice that the word accuracy of the clean test data is lower

Table 33.2 Word accuracy for the Aurora 2 test sets using the multistyle acoustic model baseline

SNR (dB)	Test set A	Test set B	Test set C	Average
Clean	99.46	99.46	99.46	99.45
20	98.99	98.59	98.78	98.79
15	98.41	97.56	98.12	98.03
10	96.94	94.94	96.05	95.98
5	91.90	88.38	87.92	89.40
0	70.53	69.36	59.35	66.42
−5	30.26	33.05	25.16	29.49
Average (0–20)	91.36	89.77	88.04	90.06

for the multistyle-trained models. This is typical of multistyle training: whereas before, the clean test data was matched to the clean training data, now every type of test data has a slight mismatch.

33.2.2 Single-Utterance Retraining

Taken to the extreme, the retraining approach outlined above could be used to generate a new acoustic model for every noisy utterance encountered.

The first step would be to extract exemplar noise signals from the current noisy utterance. This is then used to artificially corrupt a clean training corpus. Finally, an utterance specific acoustic model is trained on this corrupted data. Such a model should be a good match to the current utterance, and we would expect excellent recognition performance.

Of course, this approach would only be feasible for small systems where the training data can be kept in memory and where the retraining time is small. It would certainly not be feasible for large-vocabulary speaker-independent systems.

The idea of using an utterance-specific acoustic model can be made more efficient by replacing the recognizer retraining step with a structured adaptation of the acoustic model. Each Gaussian component in the acoustic model is adapted to account for how its parameters would change in the presence of noise. This idea is the basis of the techniques such as parallel model combination (PMC) and VTS model adaptation in Sect. 33.5, where a model for noise is composed with the acoustic model for speech, to build a noisy speech model. Although they are less accurate than the brute-force method just described, they are computationally simpler.

33.2.3 Model Adaptation

In the same way that retraining the acoustic model for each utterance can provide good noise robustness, standard unsupervised adaptation techniques can be used to approximate this effect.

Speaker adaptation algorithms, such as maximum a priori (MAP) or maximum likelihood linear regression (MLLR), are good candidates for robustness adaptation. Since MAP is an unstructured method, it can offer results similar to those of matched conditions, but it requires a significant amount of adaptation data. MLLR can achieve reasonable performance with about a minute of speech for minor mismatches [33.9]. For severe mismatches, MLLR also requires a large number of transformations, which, in turn, require a larger amount of adaptation data.

In [33.10], it was shown how MLLR works on Aurora 2. That paper reports a 9.2% relative error rate reduction over the multistyle baseline, and a 25% relative error rate reduction over the clean condition baseline.

33.3 Feature Transformation and Normalization

This section demonstrates how simple *feature normalization* techniques can be used to reduce the acoustic mismatch problem. Feature normalization works by reducing the mismatch between the training and the testing data, leading to greater robustness and higher recognition accuracies.

It has been demonstrated that feature normalization alone can provide many of the benefits of noise robustness specific algorithms. In fact, some of the best results on the Aurora 2 task have been achieved with feature normalization alone [33.11]. Because these techniques are easy to implement and provide impressive results,

they should be included in every noise-robust speech recognition system.

This section covers the most common feature normalization techniques, including voice activity detection, automatic gain normalization, cepstral mean and variance normalization, cepstral histogram normalization, and cepstral filtering.

33.3.1 Feature Moment Normalization

The goal of feature normalization is to apply a transformation to the incoming observation features. This transformation should eliminate variabilities unrelated to the transcription, while reducing the mismatches between the training and the testing utterances. Even if you do not know how the ASR features have been corrupted, it is possible to normalize them to reduce the effects of the corruption.

With moment normalization, a one-to-one transformation is applied to the data, so that its statistical moments are normalized. Techniques using this approach include cepstral mean normalization, cepstral mean and variance normalization, and cepstral histogram normalization. Respectively, they try to normalize the first, the first two, and all the moments of the data. The more moments that are normalized, the more data is needed to prevent loss of relevant acoustic information.

Another type of normalization affects only the energy-like features of each frame. Automatic gain normalization (AGN) is used to ensure that the speech occurs at the same absolute signal level, regardless of the incoming level of background noise or SNR. The simplest of these AGN schemes subtracts the maximum $C0$ value from every frame for each utterance. With this method, the most energetic frame (which is likely to contain speech) gets a $C0$ value of zero, while every other frame gets a negative $C0$.

When using moment normalization, it is sometimes beneficial to use AGN on the energy-like features, and the more-general moment normalization on the rest. For each of the moment normalization techniques discussed below, the option of treating $C0$ separately with AGN is evaluated.

Cepstral Mean Normalization

Cepstral mean normalization is the simplest feature normalization technique to implement, and should be considered first. It provides many of the benefits available in the more-advanced normalization algorithms.

For our analysis, the received speech signal $x[m]$ is used to calculate a sequence of cepstral vectors $\{\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{T-1}\}$. In its basic form, cepstral mean normalization (CMN) (*Atal* [33.12]) consists of subtracting the mean feature vector $\boldsymbol{\mu}_{\boldsymbol{x}}$ from each vector $\boldsymbol{x}_t$ to obtain the normalized vector $\hat{\boldsymbol{x}}_t$:

$$\boldsymbol{\mu}_{\boldsymbol{x}} = \frac{1}{T}\sum_t \boldsymbol{x}_t \,, \tag{33.1}$$

$$\hat{\boldsymbol{x}}_t = \boldsymbol{x}_t - \boldsymbol{\mu}_{\boldsymbol{x}} \,. \tag{33.2}$$

As a result, the long-term average of any observation sequence (the first moment) is zero.

It is easy to show that CMN makes the features robust to some linear filtering of the acoustic signal, which might be caused by microphones with different transfer functions, varying distance from user to microphone, the room acoustics, or transmission channels.

To see this, consider a signal $y[m]$, which is the output of passing $x[m]$ through a filter $h[m]$. If the filter $h[m]$ is much shorter than the analysis window used to compute the cepstra, the new cepstral sequence $\{\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{T-1}\}$ will be equal to

$$\boldsymbol{y}_t = \boldsymbol{x}_t + \boldsymbol{h} \,. \tag{33.3}$$

Here, the cepstrum of the filter $\boldsymbol{h}$ is defined as the discrete cosine transform (DCT) of the log power spectrum of the filter coefficients $h[m]$:

$$\boldsymbol{h} = \mathbf{C}\big(\ln|H(\omega_0)|^2 \ldots \ln|H(\omega_M)|^2\big) \,. \tag{33.4}$$

Since the DCT is a linear operation, it is represented here as multiplication by the matrix $\mathbf{C}$.

The sample mean of the filtered cepstra is also offset by $\boldsymbol{h}$, a factor which disappears after mean normalization:

$$\boldsymbol{\mu}_{\boldsymbol{y}} = \frac{1}{T}\sum_{t=1}^{T-1} \boldsymbol{y}_t = \boldsymbol{\mu}_{\boldsymbol{x}} + \boldsymbol{h} \,, \tag{33.5}$$

$$\hat{\boldsymbol{y}}_t = \boldsymbol{y}_t - \boldsymbol{\mu}_{\boldsymbol{y}} = \hat{\boldsymbol{x}}_t \,. \tag{33.6}$$

As long as these convolutional distortions have a time constant that is short with respect to the front end's analysis window length, and does not suppress large regions of the spectrum below the noise floor (e.g., a severe low-pass filter), CMN can virtually eliminate their effects. As the filter length $h[m]$ grows, (33.3) becomes less accurate and CMN is less effective in removing the convolutional distortion. In practice, CMN can normalize the effect of different telephone channels and microphones, but fails with reverberation times that start to approach the analysis window length [33.13].

Tables 33.3 and 33.4 show the effectiveness of whole-utterance CMN on the Aurora 2 task. In the

Table 33.3 Word accuracy for Aurora 2, using cepstral mean normalization and an acoustic model trained on clean data. CMN reduces the error rate by 31% relative to the baseline in Table 33.1

Energy normalization	Normalization level	Set A	Set B	Set C	Average
CMN	Static	68.55	73.51	69.48	70.72
AGN	Static	69.46	69.84	74.15	70.55
AGN	Full	70.34	70.74	74.88	71.41
CMN	Full	68.65	73.71	69.69	70.88

Table 33.4 Word accuracy for Aurora 2, using cepstral mean normalization and an acoustic model trained on multistyle data. CMN reduces the error rate by 30% relative to the baseline in Table 33.2

Energy normalization	Normalization level	Set A	Set B	Set C	Average
CMN	Static	92.93	92.73	93.49	92.96
AGN	Static	93.15	92.61	93.52	93.01
AGN	Full	93.11	92.63	93.56	93.01
CMN	Full	92.97	92.62	93.32	92.90

best case, CMN reduces the error rate by 31% relative using a clean acoustic model, and 30% relative using a multistyle acoustic model.

Both tables compare applying CMN on the energy feature to using AGN. In most cases, using AGN is better than applying CMN on the energy term. The failure of CMN on the energy feature is most likely due to the randomness it induces on the energy of noisy speech frames. AGN tends to put noisy speech at the same level regardless of SNR, which helps the recognizer make sharp models. On the other hand, CMN will make the energy term smaller in low-SNR utterances and larger in high-SNR utterances, leading to less-effective speech models.

There are also two different stages in which CMN can be applied. One option is to use CMN on the static cepstra, before computing the dynamic cepstra. Because of the nature of CMN, this is equivalent to leaving the dynamic cepstra untouched. The other option is to use CMN on the full feature vector, after dynamic cepstra have been computed from the unnormalized static cepstra. Tables 33.3 and 33.4 both show that it is slightly better to apply the normalization to the full feature vectors.

Cepstral Variance Normalization

Cepstral variance normalization (CVN) is similar to CMN, and the two are often paired as cepstral mean and variance normalization (CMVN). CMVN uses both the sample mean and standard deviation to normalize the cepstral sequence:

$$\sigma_x{}^2 = \frac{1}{T}\sum_{t=0}^{T-1} x_t^2 - \mu_x^2 \,, \tag{33.7}$$

$$x_t = \frac{x_t - \mu_x}{\sigma_x} \,. \tag{33.8}$$

After normalization, the mean of the cepstral sequence is zero, and it has a variance of one.

Unlike CMN, CVN is not associated with addressing a particular type of distortion. It can, however, be shown empirically that it provides robustness against acoustic channels, speaker variability, and additive noise.

Tables 33.5 and 33.6 show how CMVN affects accuracy on the Aurora 2 task. Adding variance normalization to CMN reduces the error rate 8.7% relative using a clean acoustic model, and by 8.6% relative when using a multistyle acoustic model.

As with CMN, CMVN is best applied to the full feature vector, after the dynamic cepstra have been computed. Unlike CMN, the tables show that applying CMVN to the energy term is often better than using whole-utterance AGN. Because CMVN is both shifting and scaling the energy term, both the noisy speech and the noise are placed at a consistent absolute levels.

Cepstral Histogram Normalization

Cepstral histogram normalization (CHN) [33.14] takes the core ideas behind CMN and CVN, and extends them to their logical conclusion. Instead of only normalizing the first or second central moments, CHN modifies the signal such that all of its moments are normalized. As with CMN and CHN, a one-to-one transformation is independently applied to each dimension of the feature vector.

The first step in CHN is choosing a desired distribution for the data, $p_x(x)$. It is common to choose

Table 33.5 Word accuracy for Aurora 2, using cepstral mean and variance normalization and an acoustic model trained on clean data. CMVN reduces the error rate by 8.7% relative to the CMN results in Table 33.3. CMVN is much better than AGN at energy normalization, probably because it provides consistent absolute levels for both speech and noise, whereas AGN only normalizes the speech

Energy normalization	Normalization level	Set A	Set B	Set C	Average
AGN	Static	72.96	72.4	76.48	73.44
CMVN	Static	79.34	79.86	80.8	79.84
CMVN	Full	84.46	85.55	84.84	84.97
AGN	Full	72.77	72.23	77.02	73.40

Table 33.6 Word accuracy for Aurora 2, using cepstral mean and variance normalization and an acoustic model trained on multistyle data. CVMN reduces the error rate by 8.6% relative to the baseline in Table 33.4. The difference between CMVN and AGN for energy normalization is less pronounced than in Table 33.5

Energy normalization	Normalization level	Set A	Set B	Set C	Average
AGN	Static	93.34	92.79	93.62	93.18
CMVN	Static	93.33	92.57	93.24	93.01
CMVN	Full	93.80	93.09	93.70	93.50
AGN	Full	93.37	92.76	93.70	93.19

a Gaussian distribution with zero mean and unit covariance. Let $p_y(y)$ represent the actual distribution of the data to be transformed.

It can be shown that the following function $f(\cdot)$ applied to y produces features with the probability distribution function (PDF) $p_x(x)$:

$$f(y) = F_x^{-1}[F_y(y)] \,. \tag{33.9}$$

Here, $F_y(y)$ is the cumulative distribution function (CDF) of the test data. Applying $F_y(\cdot)$ to y transforms the data distribution from $p_y(y)$ to a uniform distribution. Subsequent application of $F_x^{-1}(\cdot)$ imposes a final distribution of $p_x(x)$. When the target distribution is chosen to be Gaussian as described above, the final sequence has zero mean and unit covariance, just as if CMVN were used. Additionally, every other moment would match the target Gaussian distribution.

Whole-utterance Gaussianization is easy to implement by applying (33.9) independently to each feature dimension.

First, the data is transformed using (33.10) so it has a uniform distribution. The summation counts how many frames have the i-th dimension of $\boldsymbol{y}$ less than the value in frame m, and divides by the number of frames. The resulting sequence of $y_i'[m]$ has a uniform distribution between zero and one:

$$y_i'[m] = \frac{1}{M} \sum_{m'=1}^{M} 1\big(y_i[m'] < y_i[m]\big) \,. \tag{33.10}$$

The second and final step consists of transforming $y_i'[m]$ so that it has a Gaussian distribution. This can be accomplished, as in (33.11), using an inverse Gaussian CDF G_x^{-1}:

$$y_i^{\text{CHN}}[m] = G_x^{-1}\big(y_i'[m]\big) \,. \tag{33.11}$$

Tables 33.7 and 33.8 show the results of applying CHN to the Aurora 2 task. As with CMVN, it is better to apply the normalizing transform to a full feature vector, and to avoid the use of a separate AGN step. In the end, the results are not significantly better than CMVN.

Analysis of Feature Normalization

When implementing feature normalization, it is very important to use enough data to support the chosen technique. In general, with stronger normalization algorithms, it is necessary to process longer segments of speech.

As an example, let us analyze the effect of CMN on a short utterance. Consider an utterance contains a single phoneme, such as the fricative /s/. The mean $\boldsymbol{\mu_x}$ will be very similar to the frames in this phoneme, since /s/ is quite stationary. Thus, after normalization, $\boldsymbol{\mu_x} \approx 0$. A similar result will happen for other fricatives, which means that it would be impossible to distinguish these ultrashort utterances, and the error rate will be very high. If the utterance contains more than one phoneme but is still short, the problem is not insurmountable, but the confusion among phonemes is still higher than if no CMN had been applied.

Table 33.7 Word accuracy for Aurora 2, using cepstral histogram normalization and an acoustic model trained on clean data. As with CMVN, CHN is much better than AGN at energy normalization. The CHN results on Aurora 2 are similar to the CMVN results presented in Table 33.5

Energy normalization	Normalization level	Set A	Set B	Set C	Average
AGN	Static	70.34	71.14	73.76	71.34
CHN	Static	82.49	84.06	83.66	83.35
CHN	Full	83.64	85.03	84.59	84.39
AGN	Full	69.75	70.23	74.25	70.84

Table 33.8 Word accuracy for Aurora 2, using cepstral histogram normalization and an acoustic model trained on multistyle data. The CHN results on Aurora 2 are similar to the CMVN results presented in Table 33.8

Energy normalization	Normalization level	Set A	Set B	Set C	Average
AGN	Static	93.19	92.63	93.45	93.02
CHN	Static	93.17	92.69	93.04	92.95
CHN	Full	93.61	93.21	93.49	93.43
AGN	Full	93.29	92.73	93.74	93.16

If test utterances are too short to support the chosen normalization technique, degradation will be most apparent in the clean-speech recognition results. CMVN and CHN, in particular, can significantly degrade the accuracy of the clean speech tests in Aurora 2. In cases where there is not enough data to support CMN, *Rahim* has shown [33.15] that using the recognizer's acoustic model to estimate a maximum-likelihood mean normalization is superior to conventional CMN.

Empirically, it has been found that CMN does not degrade the recognition rate on utterances from the same acoustical environment, as long as there are at least four seconds of speech frames available. CMVN and CHN require even longer segments of speech.

As we have seen, CMN can provide robustness against additive noise. It is also effective in normalizing acoustic channels. For telephone recordings, where each call has a different frequency response, the use of CMN has been shown to provide as much as 30% relative decrease in error rate. When a system is trained on one microphone and tested on another, CMN can provide significant robustness.

Interestingly, it has been found in practice that the error rate for utterances within the same environment can actually be somewhat lower. This is surprising, given that there is no mismatch in channel conditions. One explanation is that, even for the same microphone and room acoustics, the distance between the mouth and the microphone varies for different speakers, which causes slightly different transfer functions. In addition, the cepstral mean characterizes not only the channel transfer function, but also the average frequency response of different speakers. By removing the long-term speaker average, CMN can act as sort of speaker normalization.

One drawback of CMN, CMVN, and CHN is that they do not discriminate between nonspeech and speech frames in computing the utterance mean. As a result, the normalization can be affected by the ratio of speech to nonspeech frames. For instance, the mean cepstrum of an utterance that has 90% nonspeech frames will be significantly different from one that contains only 10% nonspeech frames. As a result, the speech frames will be transformed inconsistently, leading to poorer acoustic models and decreased recognition accuracy.

An extension to CMN that addresses this problem consists in computing different means for noise and speech [33.16]:

$$\boldsymbol{h}^{j+1} = \frac{1}{N_\mathrm{s}} \sum_{t \in q_\mathrm{s}} \boldsymbol{x}_t - \boldsymbol{m}_\mathrm{s} \,, \tag{33.12}$$

$$\boldsymbol{n}^{j+1} = \frac{1}{N_\mathrm{n}} \sum_{t \in q_\mathrm{n}} \boldsymbol{x}_t - \boldsymbol{m}_\mathrm{n} \,, \tag{33.13}$$

i. e., the difference between the average vector for speech frames in the utterance and the average vector $\boldsymbol{m}_\mathrm{s}$ for speech frames in the training data, and similarly for the noise frames $\boldsymbol{m}_\mathrm{n}$. Speech/noise discrimination could be done by classifying frames into speech frames and noise frames, computing the average cepstra for each, and subtracting them from the average in the training data. This procedure works well as long as the speech/noise classification is accurate. This is best done by the recognizer,

since other speech detection algorithms can fail in high background noise.

33.3.2 Voice Activity Detection

A voice activity detection (VAD) algorithm can be used to ensure that only a small, consistent percentage of the frames sent to the speech recognizer are nonspeech frames. These algorithms work by finding and eliminating any contiguous segments of nonspeech audio in the input.

VAD is an essential component in any complete noise-robust speech recognition system. It helps to normalize the percentage of nonspeech frames in the utterance, which, as discussed above, helps CMN perform better. It also directly reduces the number of extra words, or *insertion errors* produced by the recognition system. Because these nonspeech frames are never sent to the speech recognizer, they can not be mistaken for speech. As a side-effect, because the decoder does not need to process the eliminated frames, the overall recognition process is more efficient.

Most standard databases, such as Aurora 2, have been presegmented to include only a short pause before and after each utterance. As a result, the benefits of using VAD are not apparent on that data. Other tasks, such as Aurora 3, include longer segments of noise before and after the speech, and need a good VAD for optimal performance. For example, [33.17] shows how a VAD can significantly improve performance on the Aurora 3 task.

When designing a VAD, it is important to notice that the cost of making an error is not symmetric. If a nonspeech frame is mistakenly labeled as speech, the recognizer can still produce a good result because the silence hidden Markov model (HMM) may take care of it. On the other hand, if some speech frames are lost, the recognizer cannot recover from this error.

Some VAD use a single feature, such as energy, together with a threshold. More-sophisticated systems use log-spectra or cepstra, and make decisions based on Gaussian mixture models (GMMs) or neural networks. They can leverage acoustic features that the recognizer may not, such as pitch, zero crossing rate, and duration. As a result, a VAD can do a better job than the general recognition system at rejecting nonspeech segments of the signal.

Another common way to reduce the number of insertion errors is to tune the balance of insertion and deletion errors with the recognizer's *insertion penalty* parameter. In a speech recognition system, the insertion penalty is a fixed cost incurred for each recognized word. For a given set of acoustic observations, increasing this penalty causes fewer words to be recognized. In practice, the number of insertion errors can be reduced significantly while only introducing a moderate number of deletion errors.

33.3.3 Cepstral Time Smoothing

CMN, as originally formulated, requires a complete utterance to compute the cepstral mean; thus, it cannot be used in a real-time system, and an approximation needs to be used. In this section we discuss a modified version of CMN that can address this problem, as well as a set of cepstral filtering techniques that attempt to do the same thing.

Because CMN removes any constant bias from the cepstral time series, it is equivalent to a high-pass filter with a cutoff frequency arbitrarily close to zero. This insight suggests that other types of high-pass filters may also be used. One that has been found to work well in practice is the exponential filter, so the cepstral mean $\boldsymbol{\mu}_{\boldsymbol{x}}[m]$ is a function of time:

$$\boldsymbol{\mu}_{\boldsymbol{x}}[m] = \alpha \boldsymbol{x}[m] + (1-\alpha)\boldsymbol{\mu}_{\boldsymbol{x}}[m-1]\,, \tag{33.14}$$

where α is chosen so that the filter has a time constant of at least 5 s of speech. For example, when the analysis frame rate is 100 frames per second, an α of 0.999 creates a filter with a time constant of almost 7 s.

This idea of using a filter to normalizing a sequence of cepstral coefficients is quite powerful, and can be extended to provide even better results.

In addition to a high-pass CMN-like filter, it is also beneficial to add a low-pass component to the cepstral filter. This is because the rate of change of the speech spectrum over time is limited by human physiology, but the interfering noise components are not. Abrupt spectral changes are likely to contain more noise than speech. As a result, disallowing the cepstra from changing too quickly increases their effective SNR. This is the central idea behind relative spectral (RASTA) and autoregressive moving average (ARMA) filtering.

The relative spectral processing or RASTA [33.18] combines both high- and low-pass cepstral filtering into a single noncausal infinite impulse response (IIR) transfer function:

$$H(z) = 0.1(z^4)\frac{2 + z^{-1} - z^{-3} - 2z^{-4}}{1 - 0.98z^{-1}}\,. \tag{33.15}$$

As in CMN, the high-pass portion of the filter is expected to alleviate the effect of convolutional noise introduced in the channel. The low-pass filtering helps to

smooth some of the fast frame-to-frame spectral changes present. Empirically, it has been shown that the RASTA filter behaves similarly to the real-time implementation of CMN, albeit with a slightly higher error rate.

ARMA filtering is similar to RASTA, in that a linear time invariant (LTI) filter is applied separately to each cepstral coefficient. The following equation:

$$H(z) = (z^2)\frac{1+z^{-1}+z^{-2}}{5-z^{-1}-z^{-2}} \tag{33.16}$$

is an example of a second-order ARMA filter. Unlike RASTA, ARMA is purely a low-pass filter. As a result, ARMA should be used in conjunction with an additional explicit CMN operation.

It was shown in [33.11] how this simple technique can do better than much more-complex robustness schemes. In spite of its simplicity, those results were the best of their time.

33.3.4 SPLICE – Normalization Learned from Stereo Data

The SPLICE technique was first proposed [33.19] as a brute-force solution to the acoustic mismatch problem. Instead of blindly transforming the data as CMN, CHN, or RASTA, SPLICE learns the joint probability distribution for noisy speech and clean speech, and uses it to map each received cepstra into a clean estimate. Like CHN, SPLICE is a nonlinear transformation technique. However, whereas CHN implicitly assumes the features are uncorrelated, SPLICE learns and uses the correlations naturally present in speech features.

The SPLICE transform is built from a model of the joint distribution of noisy cepstra $\boldsymbol{y}$ and clean cepstra $\boldsymbol{x}$. The model is a Gaussian mixture model containing K mixture components:

$$p(\boldsymbol{y},\boldsymbol{x}) = \sum_{k=1}^{K} p(\boldsymbol{x}|\boldsymbol{y},k)p(\boldsymbol{y},k)\,. \tag{33.17}$$

The distribution $p(\boldsymbol{y},k)$ is itself a Gaussian mixture model on $\boldsymbol{y}$, which takes the form

$$p(\boldsymbol{y},k) = p(\boldsymbol{y}|k)p(k) = N(\boldsymbol{y};\mu_k,\sigma_k)p(k)\,. \tag{33.18}$$

The conditional distribution $p(\boldsymbol{x}|\boldsymbol{y},k)$ predicts the clean feature value given a noisy observation and a Gaussian component index k:

$$p(\boldsymbol{x}|\boldsymbol{y},k) = N(\boldsymbol{x};A_k\boldsymbol{y}+b_k,\Gamma_k)\,. \tag{33.19}$$

Due to their effect on predicting $\boldsymbol{x}$ from $\boldsymbol{y}$, the matrix A_k is referred to as the *rotation matrix*, and the vector b_k is called the *offset vector*. The matrix Γ_k represents the error incurred in the prediction.

Even though the relationship between $\boldsymbol{x}$ and $\boldsymbol{y}$ is nonlinear, this conditional linear prediction is sufficient. Because the GMM $p(\boldsymbol{y},k)$ effectively partitions the noisy acoustic space into K regions, each $p(\boldsymbol{x}|\boldsymbol{y},k)$ only needs to be accurate in one of these regions.

The SPLICE transform is derived from the joint distribution $p(\boldsymbol{x},\boldsymbol{y},k)$ by finding the expected value of the clean speech $\boldsymbol{x}$, given the current noisy observation $\boldsymbol{y}$. This approach finds the minimum mean-squared error (MMSE) estimate for x under the model, an approach pioneered by the codeword-dependent cepstral normalization (CDCN) [33.20] and multivariate-gaussian-based cepstral normalization (RATZ) [33.21] algorithms:

$$\begin{aligned}\hat{\boldsymbol{x}}_{\text{MMSE}} &= E\{\boldsymbol{x}|\boldsymbol{y}\} = \sum_{k=1}^{K} E\{\boldsymbol{x}|\boldsymbol{y},k\}p(k|\boldsymbol{y}) \\ &= \sum_{k=1}^{K}(A_k\boldsymbol{y}+b_k)p(k|\boldsymbol{y})\,,\end{aligned} \tag{33.20}$$

where the posterior probability $p(k|\boldsymbol{y})$ is given by

$$p(k|\boldsymbol{y}) = \frac{p(\boldsymbol{y},k)}{\sum_{k'=1}^{K} p(\boldsymbol{y},k')}\,. \tag{33.21}$$

Another option is to find an approximate maximum likelihood estimate for $\boldsymbol{x}$ under the model, which is similar to the fixed codeword-dependent cepstral normalization algorithm algorithm [33.20]. A good approximate solution to

$$\hat{\boldsymbol{x}}_{\text{ML}} = \max_x p(\boldsymbol{x}|\boldsymbol{y}) \tag{33.22}$$

is

$$\hat{\boldsymbol{x}}_{\text{ML}} \approx A_{\hat{k}}\boldsymbol{y}+b_{\hat{k}}\,, \tag{33.23}$$

where

$$\hat{k} = \arg\max_k p(\boldsymbol{y},k)\,. \tag{33.24}$$

Whereas the approximate maximum-likelihood (ML) estimate above involves a single affine transformation, the MMSE solution requires K transformations. Even though the MMSE estimate produces more-accurate recognition results, the approximate ML estimate may be substituted when the additional computational cost is prohibitive.

Another popular method for reducing the additional computational cost is to replace the learned rotation matrix A_k with the identity matrix I. This produces systems that are more efficient, with only a modest degradation in performance [33.19].

A number of different algorithms [33.20, 21] have been proposed that vary in how the parameters μ_k, Σ_k, A_k, b_k, and Γ_k are estimated.

If stereo recordings are available from both the clean signal and the noisy signal, then we can estimate μ_k and Σ_k by fitting a mixture Gaussian model to $\boldsymbol{y}$ using standard maximum-likelihood training techniques. Then A_k, b_k, and Γ_k can be estimated directly by linear regression of $\boldsymbol{x}$ and $\boldsymbol{y}$. The FCDCN algorithm [33.20, 22] is a variant of this approach when it is assumed that $\Sigma_k = \sigma^2 I$, $\Gamma_k = \gamma^2 I$, and $A_k = I$, so that μ_k and b_k are estimated through a vector quantization (VQ) procedure and b_k is the average difference $(\boldsymbol{y} - \boldsymbol{x})$ for vectors $\boldsymbol{y}$ that belong to mixture component k.

Often, stereo recordings are not available and we need other means of estimating the parameters μ_k, Σ_k, A_k, b_k, and Γ_k. CDCN [33.22] and VTS enhancement [33.21] are examples of algorithms that use a model of the environment (Sect. 33.4). This model defines a nonlinear relationship between $\boldsymbol{x}$, $\boldsymbol{y}$ and the environmental parameters $\boldsymbol{n}$ for the noise. The CDCN method also uses an MMSE approach where the correction vector is a weighted average of the correction vectors for all classes. Other methods that do not require stereo recordings or a model of the environment are presented in [33.21].

Recent improvements in training the transformation parameters have been proposed using discriminative training [33.23–25]. In this case, the noisy-speech GMM is developed from the noisy training data, or from a larger acoustic model developed from that data. The correction parameters are then initialized to zero, which corresponds to the identity transformation. Subsequently, they are trained to maximize a discriminative criterion, such as minimum classification error (MCE) [33.24], maximum mutual information (MMI) [33.23], or minimum phone error (MPE) [33.25]. It has been shown that this style of training produces superior results to the two-channel MMSE approach, and can even increase accuracy in noise-free test cases. The main disadvantage of this approach is that it is easy to overtrain the transformation, which can actually reduce robustness of the system to noise types that do not occur in the training set. This can be easily avoided by the customary use of regularization and separate training, development, and test data.

Although the SPLICE transform is discussed here, there are many alternatives available. The function to map from $\boldsymbol{y}$ to $\boldsymbol{x}$ can be approximated with a neural network [33.26], or as a mixture of Gaussians as in probabilistic optimum filtering (POF) [33.27], FCDCN [33.28], RATZ [33.21], and stochastic vector mapping (SVM) [33.24].

33.4 A Model of the Environment

Sections 33.2 and 33.3 described techniques that address the problem of additive noise by blindly reducing the acoustic mismatch between the acoustic model and the expected test data. These techniques are powerful and general, and are often used to solve problems other than additive noise. However, the more-effective solutions generally require more data to operate properly.

In this section, we use knowledge of the nature of the degradation to derive the relationship between the clean and observed signals in the power-spectrum, log-filterbank, and cepstral domains. Later, Sects. 33.5 and 33.6 show how several related methods leverage this model to produce effective noise robust speech recognition techniques.

In the acoustic environment, the clean speech signal coexists with many other sound sources. They mix linearly in the air, and a mixture of all these signals is picked up by the microphone. For our purposes, the clean speech signal that would have existed in the absence of noise is denoted by the symbol $x[m]$, and all of the other noises that are picked up by the microphone are represented by the symbol $n[m]$. Because of the linear mixing, the observed signal $y[m]$ is simply the sum of the clean speech and noise:

$$y[m] = x[m] + n[m] \,. \tag{33.25}$$

Unfortunately, the additive relationship of (33.25) is destroyed by the nonlinear process of extracting cepstra from $y[m]$. Figure 33.2 demonstrates the path that the noisy signal takes on its way to becoming a cepstral feature vector observation vector. It consists of dividing the received signal into frames, performing a frequency analysis and warping, applying a logarithmic compression, and finally a decorrelation and dimensionality reduction.

The first stage of feature extraction is framing. The signal is split into overlapping segments of about 25 ms each. These segments are short enough that, within each frame of data, the speech signal is approximately stationary. Each frame is passed through a discrete Fourier transform (DFT), where the time-domain signal becomes a complex-valued function of discrete fre-

quency k. Concentrating our analysis on a single frame of speech $Y[k]$, clean speech and noise are still additive:

$$Y[k] = X[k] + N[k] \,. \tag{33.26}$$

The next processing step is to turn the observed complex spectra into real-valued power spectra, through the application of a magnitude-squared operation. The power spectrum of the observed signal is

$$|Y[k]|^2 = |X[k]|^2 + |N[k]|^2 + 2|X[k]||N[k]| \cos\theta \,, \tag{33.27}$$

a function of the power spectrum of the clean speech and of the noise, as well as a *cross-term*. This cross-term is a function of the clean speech and noise magnitudes, as well as their relative phase θ. When the clean speech and noise are uncorrelated, the expected value of the cross-term is zero:

$$E\{X[k]N^*[k]\} = 0 \,. \tag{33.28}$$

However, for a particular frame it can have a considerable magnitude.

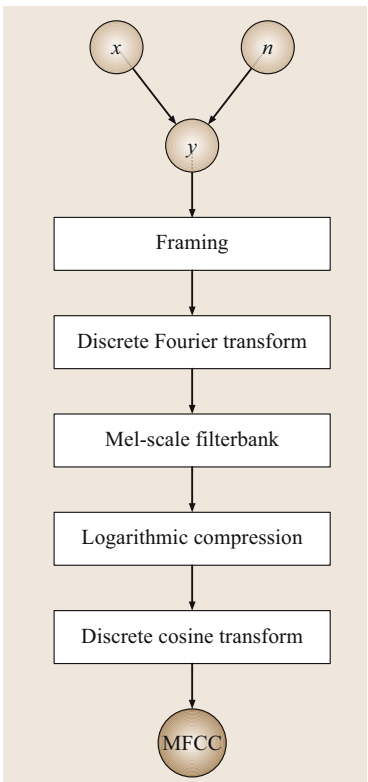


Fig. 33.2 Clean speech x and environmental noise n mix to produce the noisy signal y, which is turned into mel-frequency cepstral coefficients (MFCC) through a sequence of processing steps

It is uncommon to pass the Fourier spectra directly to the speech recognition system. Instead, it is standard to apply a mel-frequency filterbank and a logarithmic compression to create log mel-frequency filterbank (LMFB) features. The mel-frequency filterbank implements dimensionality reduction and frequency warping as a linear projection of the power spectrum. The relationship between the i-th LMFB coefficient y_i and the observed noisy spectra $Y[k]$ is given by

$$y_i = \ln\left(\sum_k w_k^i |Y[k]|^2\right) \,, \tag{33.29}$$

where the scalar w_k^i is the k-th coefficient of the i-th filter in the filterbank.

Figure 33.3 reveals a typical structure of the matrix $\boldsymbol{W}$ containing the scalars w_k^i. It compresses 128 FFT bins into 23 mel-frequency spectral features with a resolution that varies over frequency. At low frequencies, high resolution is preserved by using only a small number of FFT bins for each mel-frequency feature. As the Fourier frequency increases, more FFT bins are used.

Using (33.29) and (33.27), we can deduce the relationship among the LMFB for clean speech, noise, and noisy observation. The noisy LMFB energies are a function of the two unobserved LMFB, and a cross-term that depends on a third nuisance parameter α_i:

$$\exp(y_i) = \exp(x_i) + \exp(n_i) + 2\alpha_i \exp\left(\frac{x_i + n_i}{2}\right) \,, \tag{33.30}$$

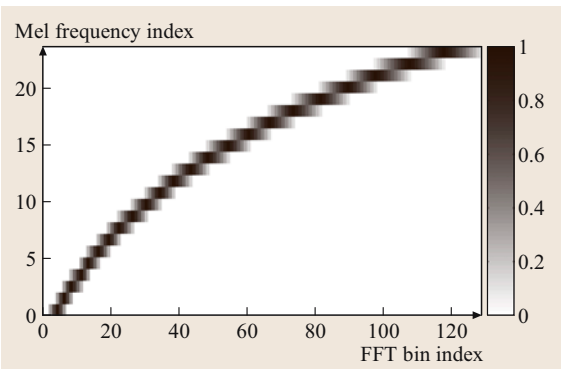


Fig. 33.3 A graphical representation of the mel-frequency filterbank used to compute MFCC features. This filterbank compresses 128 spectral bins into 23 mel-frequency coefficients

where

$$\alpha_i = \frac{\sum_k w_k^i |X[k]N[k]| \cos\theta_k}{\sqrt{\sum_k w_k^i |X[k]|^2}\sqrt{\sum_k w_k^i |N[k]|^2}} . \quad (33.31)$$

As a consequence of this model, when we observe y_i there are actually *three* unobserved random variables. The first two are obvious: the clean log spectral energy and the noise log spectral energy that would have been produced in the absence of mixing. The third variable α_i accounts for the unknown phase between the two sources.

If the magnitude spectra are assumed constant over the bandwidth of a particular filterbank, the definition of α_i collapses to a weighted sum of several independent random variables

$$\alpha_i = \frac{\sum_k w_k^i \cos\theta_k}{\sum_j w_j^i} . \quad (33.32)$$

According to the central limit theorem, a sum of many independent random variables will tend to be normally distributed. The number of effective terms in (33.32) is controlled by the width of the i-th filterbank. Since filterbanks with higher center frequencies have wider bandwidths, they should be more nearly Gaussian. Figure 33.4 shows the true distributions of α for a range of filterbanks. They were estimated from a joint set of noise and clean speech, and noisy speech taken from the Aurora 2 training data by solving (33.30) for the unknown α_i. As expected, because the higher-frequency

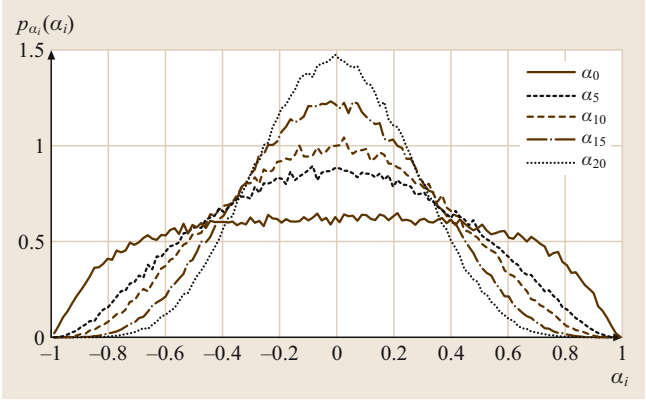


Fig. 33.4 An estimation of the true distribution of α using (33.30) and data from the Aurora 2 corpus. Higher numbered filterbanks cover more frequency bins, are more nearly Gaussian, and have smaller variances

higher-bandwidth filters include more FFT bins, they produce distributions that are more nearly Gaussian.

After some algebraic manipulation, it can be shown that

$$y_i = x_i + \ln\left[1 + \exp(n_i - x_i) + 2\alpha_i \exp\left(\tfrac{n_i - x_i}{2}\right)\right] . \quad (33.33)$$

Many current speech enhancement algorithms ignore the cross-term entirely, as in

$$y_i = x_i + \ln[1 + \exp(n_i - x_i)] . \quad (33.34)$$

This is entirely appropriate in situations where the observed frequency component is dominated by either noise or speech. In these cases, the cross-term is indeed negligible. But, in cases where the speech and noise components have similar magnitudes, the cross-term can have a considerable effect on the observation.

To create cepstral features from these LMFB features, apply a discrete cosine transform. The j-th cepstral coefficient is calculated with the following sum, where c_{ij} are the DCT coefficients:

$$y_j^{\text{MFCC}} = \sum_i c_{ji} y_i^{\text{LMFB}} . \quad (33.35)$$

By convention, this transform includes a truncation of the higher-order DCT coefficients, a process historically referred to as *cepstral liftering*. For example, the Aurora 2 system uses 23 LMFB and keeps only the first 13 cepstral coefficients.

Due to this cepstral truncation, the matrix $\mathbf{C}$ created from the scalars c_{ij} is not square and cannot be inverted. But, we can define a right-inverse matrix $\mathbf{D}$ with elements d_{ij}, such that $\mathbf{CD} = \mathbf{I}$. That is, if $\mathbf{D}$ is applied to a cepstral vector, an approximate spectral vector is produced, from which the projection $\mathbf{C}$ will recreate the original cepstral vector. Using $\mathbf{C}$ and $\mathbf{D}$, (33.34) can be expressed for cepstral vectors $\boldsymbol{x}$, $\boldsymbol{n}$, and $\boldsymbol{y}$ as:

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{g}(\boldsymbol{x} - \boldsymbol{n}) , \quad (33.36)$$

$$\boldsymbol{g}(z) = \mathbf{C}\ln[1 + \exp(-\mathbf{D}z)] . \quad (33.37)$$

Although $\mathbf{CD} = \mathbf{I}$, it is not the case that $\mathbf{DC} = \mathbf{I}$. In particular, (33.37) is not exactly correct as it needs an extra term $\bar{\boldsymbol{D}}\bar{\boldsymbol{z}}$ that contains the missing columns from $\mathbf{D}$ and higher-order cepstral coefficients from $\boldsymbol{z}$. If only the truncated cepstrum $\boldsymbol{z}$ is available, then $\bar{\boldsymbol{D}}\bar{\boldsymbol{z}}$ is a random error term that is generally ignored.

In (33.37), the nonlinearity $\boldsymbol{g}$ was introduced, which maps the signal-to-noise ratio $(\boldsymbol{x} - \boldsymbol{n})$ into the difference between clean and noisy speech $(\boldsymbol{y} - \boldsymbol{x})$. When the noise

cepstrum $\boldsymbol{n}$ has significantly less energy than the speech vector $\boldsymbol{x}$, the function $\boldsymbol{g}(\boldsymbol{x}-\boldsymbol{n})$ approaches zero, and $\boldsymbol{y} \approx \boldsymbol{x}$. Conversely, when noise dominates speech, $\boldsymbol{g}(\boldsymbol{x}-\boldsymbol{n}) \approx \boldsymbol{n}-\boldsymbol{x}$, and $\boldsymbol{y} \approx \boldsymbol{n}$.

33.5 Structured Model Adaptation

This section compares two popular methods for structured model adaptation: log-normal parallel model combination and vector Taylor-series adaptation. Both can achieve good adaptation results with only a small amount of data.

By using a set of clean-speech acoustic models and a noise model, both methods approximate the model parameters that would have been obtained by training with corrupted speech.

33.5.1 Analysis of Noisy Speech Features

Figures 33.5 and 33.6 depict how additive noise can distort the spectral features used in ASR systems. In both figures, the simulated clean speech x and noise n are assumed to follow Gaussian distributions:

$$p_x(x) = N(x; \mu_x, \sigma_x) ,$$
$$p_n(n) = N(n; \mu_n, \sigma_n) .$$

After mixing, the noisy speech y distribution is a distorted version of its clean speech counterpart. It is usually shifted, often skewed, and sometimes bimodal. It is clear that a pattern recognition system trained on clean speech will be confused by noisy speech input.

In Fig. 33.5, the clean speech and noise mix to produce a bimodal distribution. We fix $\mu_n = 0\,\text{dB}$, since it is only a relative level, and set $\sigma_n = 2\,\text{dB}$, a typical value. We also set $\mu_x = 25\,\text{dB}$ and see that the resulting distribution is bimodal when σ_x is very large. Fortunately, for modern speech recognition systems that have many Gaussian components, σ_x is never that large and the resulting distribution is often unimodal.

Figure 33.6 demonstrates the more-common skew and offset distortions. Again, the noise parameters are $\mu_n = 0\,\text{dB}$ and $\sigma_n = 2\,\text{dB}$, but a more-realistic value for $\sigma_x = 5\,\text{dB}$ is used. We see that the distribution is always unimodal, although not necessarily symmetric, particularly for low SNR ($\mu_x - \mu_n$).

33.5.2 Log-Normal Parallel Model Combination

Parallel model combination (PMC) [33.29] is a general method to obtain the distribution of noisy speech given the distribution of clean speech and noise as mixtures of Gaussians.

As discussed in Sect. 33.5.1, even if the clean speech and noise cepstra follow Gaussian distributions, the noisy speech will not be Gaussian. The log-normal PMC method nevertheless assumes that the resulting noisy observation is Gaussian. It transforms the clean speech and noise distributions into the linear spectral domain, mixes them, and trans-

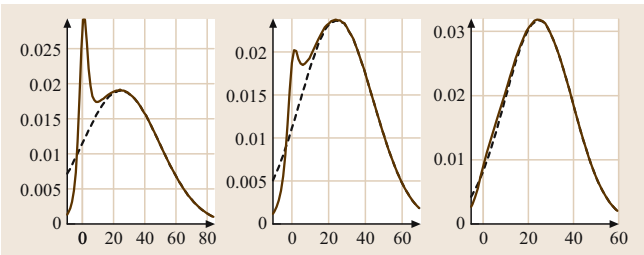


Fig. 33.5 Distributions of the corrupted log-spectra y (*solid lines*) from uncorrupted log-spectra x (*dashed lines*) using simulated data and (33.27). The distribution of the noise log-spectrum n is Gaussian with mean 0 dB and standard deviation of 2 dB. The distribution of the clean log-spectrum x is Gaussian with mean 25 dB and standard deviations of 25, 20, and 15 dB, respectively (the x-axis is expressed in dB). The first two distributions are bimodal, whereas the 15 dB case is more approximately Gaussian

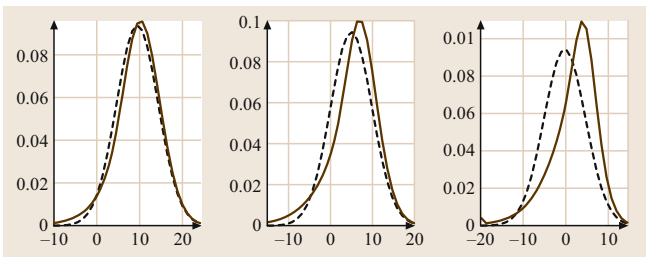


Fig. 33.6 Distributions of the corrupted log-spectra y (*solid lines*) from uncorrupted log-spectra x (*dashed lines*) using simulated data and (33.27). The distribution of the noise log-spectrum n is Gaussian with mean 0 dB and standard deviation of 2 dB. The distribution of the clean log-spectrum is Gaussian with standard deviation of 5 dB and means of 10, 5, and 0 dB, respectively. As the average SNR decreases, the Gaussian experiences more shift and skew

forms them back into a distribution on noisy speech cepstra.

If the mean and covariance matrix of the cepstral noise vector $\boldsymbol{n}$ are given by $\mu_{\boldsymbol{n}}^{\mathrm{c}}$ and $\Sigma_{\boldsymbol{n}}^{\mathrm{c}}$, respectively, its mean and covariance matrix in the log spectral domain can be approximated by

$$\mu_{\boldsymbol{n}}^{\mathrm{l}} = \mathbf{D}\mu_{\boldsymbol{n}}^{\mathrm{c}} , \tag{33.38}$$

$$\Sigma_{\boldsymbol{n}}^{\mathrm{l}} = \mathbf{D}\Sigma_{\boldsymbol{n}}^{\mathrm{c}}\mathbf{D}^{\mathrm{T}} . \tag{33.39}$$

As in Sect. 33.4, $\mathbf{D}$ is a right inverse for the noninvertible cepstral rotation $\mathbf{C}$.

In the linear domain $\boldsymbol{N} = \mathrm{e}^{\boldsymbol{n}}$, the noise distribution is log-normal, with a mean vector $\boldsymbol{\mu}_{\boldsymbol{N}}$ and covariance matrix $\Sigma_{\boldsymbol{N}}$ given by

$$\mu_{\boldsymbol{N}}[i] = \exp\left(\mu_{\boldsymbol{N}}^{\mathrm{l}}[i] + \frac{1}{2}\Sigma_{\boldsymbol{N}}^{\mathrm{l}}[i,i]\right) , \tag{33.40}$$

$$\Sigma_{\boldsymbol{N}}[i,j] = \mu_{\boldsymbol{N}}[i]\mu_{\boldsymbol{N}}[j]\left[\exp\left(\Sigma_{\boldsymbol{N}}^{\mathrm{l}}[i,j]\right) - 1\right] . \tag{33.41}$$

As a result, we have the exact parameters for the distribution of noise features in the linear spectral domain. The cepstral Gaussian distribution for the clean speech can be transformed from $\mu_{\boldsymbol{x}}^{\mathrm{c}}$ and $\Sigma_{\boldsymbol{x}}^{\mathrm{c}}$ to $\mu_{\boldsymbol{X}}$ and $\Sigma_{\boldsymbol{X}}$ using expressions similar to (33.38) through (33.41).

Using the basic assumption that the noise and speech waveforms are additive, the spectral vector $\boldsymbol{Y}$ is given by $\boldsymbol{Y} = \boldsymbol{X} + \boldsymbol{N}$. Without any approximation, the mean and covariance of $\boldsymbol{Y}$ is given by

$$\mu_{\boldsymbol{Y}} = \mu_{\boldsymbol{X}} + \mu_{\boldsymbol{N}} , \tag{33.42}$$

$$\Sigma_{\boldsymbol{Y}} = \Sigma_{\boldsymbol{X}} + \Sigma_{\boldsymbol{N}} . \tag{33.43}$$

Although the sum of two log-normal distributions is not log-normal, the *log-normal approximation* [33.29] consists in assuming that $\boldsymbol{Y}$ is log-normal. In this case we can apply the inverse formulae of (33.40) and (33.41) to obtain the mean and covariance matrix in the log-spectral domain:

$$\Sigma_{\boldsymbol{y}}^{\mathrm{l}} \approx \ln\left(\frac{\Sigma_{\boldsymbol{Y}}[i,j]}{\mu_{\boldsymbol{Y}}[i]\mu_{\boldsymbol{Y}}[j]} + 1\right) , \tag{33.44}$$

$$\mu_{\boldsymbol{y}}^{\mathrm{l}}[i] \approx \ln(\mu_{\boldsymbol{y}}[i]) - \frac{1}{2}\ln\left(\frac{\Sigma_{\boldsymbol{Y}}[i,j]}{\mu_{\boldsymbol{Y}}[i]\mu_{\boldsymbol{Y}}[j]} + 1\right) , \tag{33.45}$$

and finally return to the cepstrum domain applying the inverse of (33.38) and (33.39):

$$\mu_{\boldsymbol{y}}^{\mathrm{c}} = \mathbf{C}\mu_{\boldsymbol{y}}^{\mathrm{l}} , \tag{33.46}$$

$$\Sigma_{\boldsymbol{y}}^{\mathrm{c}} = \mathbf{C}\Sigma_{\boldsymbol{y}}^{\mathrm{l}}\mathbf{C}^{\mathrm{T}} . \tag{33.47}$$

The log-normal approximation cannot be used directly for the delta and delta–delta cepstrum. Another variant that can be used in this case and is more accurate than the log-normal approximation is the *data-driven parallel model combination* (DPMC) [33.29]. DPMC uses Monte Carlo simulation to draw random cepstrum vectors from both the clean-speech HMM and noise distribution to create cepstrum of the noisy speech by applying (33.36) to each sample point. The mean and covariance of these simulated noisy cepstra are then used as adapted HMM parameters. In that respect, it is similar to other model adaptation schemes, but not as accurate as the matched condition training from Sect. 33.2.1 because the distribution is only an approximation.

Although it does not require a lot of memory, DPMC carries with it large computational burden. For each transformed Gaussian component, the recommended number of simulated training vectors is at least 100 [33.29]. A way of reducing the number of random vectors needed to obtain good Monte Carlo simulations is proposed in [33.30].

33.5.3 Vector Taylor-Series Model Adaptation

Vector Taylor-series (VTS) model adaptation is similar in spirit to the log-normal PMC in Sect. 33.5.2, but instead of a log-normal approximation it uses a first-order Taylor-series approximation of the model presented from Sect. 33.4.

This model described the relationship between the cepstral vectors $\boldsymbol{x}$, $\boldsymbol{n}$, and $\boldsymbol{y}$ of the clean speech, noise, and noisy speech, respectively:

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{g}(\boldsymbol{n} - \boldsymbol{x}) , \tag{33.48}$$

where the nonlinear function $\boldsymbol{g}(\boldsymbol{z})$ is given by

$$\boldsymbol{g}(\boldsymbol{z}) = \mathbf{C}\ln[1 + \exp(\mathbf{D}\boldsymbol{z})] . \tag{33.49}$$

As in Sect. 33.4, $\mathbf{C}$ and $\mathbf{D}$ are the cepstral rotation and its right inverse, respectively.

Moreno [33.31] first suggested the use of Taylor series to approximate the nonlinearity in (33.49), though he applies it in the spectral instead of the cepstral domain. Since then, many related techniques have appeared that build upon this core idea [33.32–37]. They mainly differ in how speech and noise are modeled, as well as which assumptions are made in the derivation of the nonlinearity to approximate [33.38].

To apply the vector Taylor-series approximation, first assume that $\boldsymbol{x}$ and $\boldsymbol{n}$ are Gaussian random vectors with means $\{\mu_{\boldsymbol{x}}, \mu_{\boldsymbol{n}}\}$ and covariance matrices $\{\Sigma_{\boldsymbol{x}}, \Sigma_{\boldsymbol{n}}\}$,

Table 33.9 Word accuracy for Aurora 2, using VTS model adaptation on an acoustic model trained on clean data

SNR (dB)	Test set A	Test set B	Test set C	Average
Clean	99.63	99.63	99.63	99.63
20	97.86	97.59	98.17	97.88
15	96.47	95.64	96.99	96.37
10	93.47	91.98	93.30	92.91
5	86.87	85.48	85.49	85.94
0	71.68	68.90	68.72	69.77
−5	36.84	33.04	39.04	35.97
Average (0–20)	89.27	87.92	88.53	88.58

and furthermore that $\boldsymbol{x}$ and $\boldsymbol{n}$ are independent. After algebraic manipulation it can be shown that the Jacobian of (33.48) with respect to $\boldsymbol{x}$ and $\boldsymbol{n}$ evaluated at $\{\boldsymbol{x}=\mu_{\boldsymbol{x}}, \boldsymbol{n}=\mu_{\boldsymbol{n}}\}$ can be expressed as:

$$\left.\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}\right|_{(\mu_{\boldsymbol{x}},\mu_{\boldsymbol{n}})} = \mathbf{G}\,, \tag{33.50}$$

$$\left.\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{n}}\right|_{(\mu_{\boldsymbol{x}},\mu_{\boldsymbol{n}})} = \mathbf{I}-\mathbf{G}\,, \tag{33.51}$$

where the matrix $\mathbf{G}$ is given by

$$\mathbf{G}=\mathbf{CFD}\,. \tag{33.52}$$

In (33.52), $\mathbf{F}$ is a diagonal matrix whose elements are given by vector $\boldsymbol{f}(\mu)$, which in turn is given by

$$\boldsymbol{f}(\mu_{\mathrm{n}}-\mu_x)=\frac{1}{1+\exp[\mathbf{D}(\mu_{\mathrm{n}}-\mu_x)]}\,. \tag{33.53}$$

Using (33.50) and (33.51) we can then approximate (33.48) by a first-order Taylor-series expansion around $\{\mu_{\mathrm{n}}, \mu_x\}$ as

$$\begin{aligned}\boldsymbol{y} \approx{}& \mu_x+\boldsymbol{g}(\mu_{\mathrm{n}}-\mu_x)+\mathbf{G}(\boldsymbol{x}-\mu_x)\\ &+(\mathbf{I}-\mathbf{G})(\boldsymbol{n}-\mu_{\mathrm{n}})\,.\end{aligned} \tag{33.54}$$

The mean of $\boldsymbol{y}$, μ_y, can be obtained from (33.54) as

$$\mu_y \approx \mu_x+\boldsymbol{g}(\mu_{\mathrm{n}}-\mu_x)\,, \tag{33.55}$$

and its covariance matrix Σ_y by

$$\Sigma_y \approx \mathbf{G}(\Sigma_x)\mathbf{G}^{\mathrm{T}}+(\mathbf{I}-\mathbf{G})\Sigma_n(\mathbf{I}-\mathbf{G})^{\mathrm{T}}\,. \tag{33.56}$$

Note that, even if Σ_x and Σ_n are diagonal, Σ_y is not. Nonetheless, it is common to make a diagonal assumption. That way, we can transform a clean HMM to a corrupted HMM that has the same functional form and use a decoder that has been optimized for diagonal covariance matrices.

To compute the means and covariance matrices of the delta and delta–delta parameters, let us take the derivative of the approximation of $\boldsymbol{y}$ in (33.54) with respect to time:

$$\frac{\partial \boldsymbol{y}}{\partial t} \approx \mathbf{G}\frac{\partial \boldsymbol{x}}{\partial t}\,, \tag{33.57}$$

so that the delta cepstrum computed through $\Delta\boldsymbol{x}_t=\boldsymbol{x}_{t+2}-\boldsymbol{x}_{t-2}$, is related to the derivative [33.39] by

$$\Delta\boldsymbol{x} \approx 4\frac{\partial \boldsymbol{x}}{\partial t}\,, \tag{33.58}$$

so that

$$\mu_{\Delta\boldsymbol{y}} \approx \mathbf{G}\mu_{\Delta\boldsymbol{x}}\,, \tag{33.59}$$

and similarly

$$\Sigma_{\Delta\boldsymbol{y}} \approx \mathbf{G}\Sigma_{\Delta\boldsymbol{x}}\mathbf{G}^{\mathrm{T}}+(\mathbf{I}-\mathbf{G})\Sigma_{\Delta\boldsymbol{n}}(\mathbf{I}-\mathbf{G})^{\mathrm{T}}\,. \tag{33.60}$$

Similarly, for the delta–delta cepstrum, the mean is given by

$$\mu_{\Delta^2\boldsymbol{y}} \approx \mathbf{G}\mu_{\Delta^2\boldsymbol{x}}\,, \tag{33.61}$$

and the covariance matrix by

$$\Sigma_{\Delta^2\boldsymbol{y}} \approx \mathbf{G}\Sigma_{\Delta^2\boldsymbol{x}}\mathbf{G}^{\mathrm{T}}+(\mathbf{I}-\mathbf{G})\Sigma_{\Delta^2\boldsymbol{n}}(\mathbf{I}-\mathbf{G})^{\mathrm{T}}\,. \tag{33.62}$$

Table 33.9 demonstrates the effectiveness of VTS model adaptation on the Aurora 2 task. For each test noise condition, a diagonal Gaussian noise model was estimated from the first and last 200 ms of all the test data. Then, each Gaussian component in the acoustic model was transformed according to the algorithm presented above to create a noise condition specific acoustic model. In theory, better results could have been obtained by estimating a new noise model from each utterance and creating an utterance specific acoustic model. However, the computational cost would be much greater.

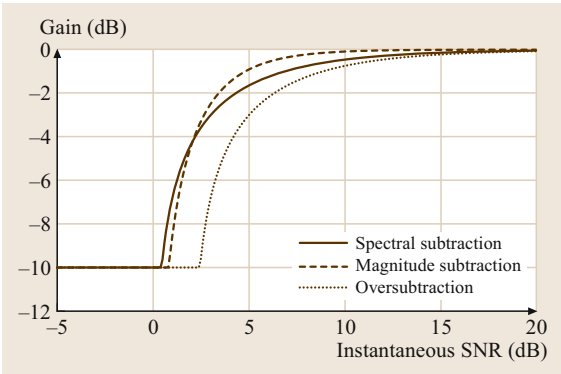


Fig. 33.7 Magnitude of the spectral subtraction filter gain as a function of the input instantaneous SNR for $A = 10$ dB, for the spectral subtraction of (33.68), magnitude subtraction of (33.71), and over-subtraction of (33.72) with $\beta = 2$ dB

Compared to the unadapted results of Table 33.1, the performance of the VTS adapted models is improved at each SNR level on every test set. At 20 dB SNR, the VTS model adaptation reduces the number of errors by 61%. Averaged over 0–20 dB, the adapted system has an average of 73% fewer errors.

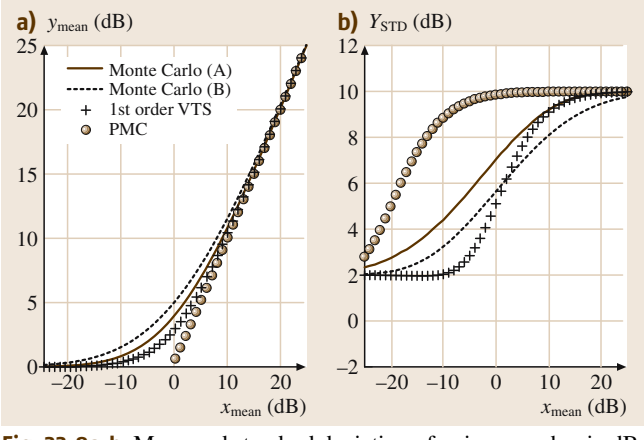


Fig. 33.8a,b Mean and standard deviation of noisy speech y in dB. The distribution of the noise log-spectrum n is Gaussian with mean 0 dB and standard deviation of 2 dB. The distribution of the clean speech log-spectrum x is Gaussian, having a standard deviation of 10 dB and a mean varying from −25 to 25 dB. The first-order VTS approximation is a better estimate for a Monte Carlo simulation of (33.27) when the cross-term is ignored **(b)**, although both VTS and lognormal PMC underestimate the standard deviation compared to when the cross-term is included **(a)**

The VTS model adaptation results also better than any feature normalization technique using clean training data from Sect. 33.3. This is a direct result of using the model from Sect. 33.4 to make better use of the available adaptation data.

33.5.4 Comparison of VTS and Log-Normal PMC

It is difficult to visualize how good the VTS approximation is, given the nonlinearity involved. To provide some insight, Figs. 33.8 and 33.9 provide a comparison between the log-normal approximation, the VTS approximation, and Monte Carlo simulations of (33.27). For simplicity, only a single dimension of the log spectral features are shown.

Figure 33.8 shows the mean and standard deviation of the noisy log-spectral energy y in dB as a function of the mean of the clean log-spectral energy x with a standard deviation of 10 dB. The log-spectral energy of the noise n is Gaussian with mean 0 dB and standard deviation 2 dB.

We see that the VTS approximation is more accurate than the log-normal approximation for the mean, especially in the region around 0 dB SNR. For the standard deviation, neither approximation is very accurate. Because the VTS models fail to account for the cross-term of (33.27), both tend to underestimate the true noisy standard deviation.

Figure 33.9 is similar to Fig. 33.8 except that the standard deviation of the clean log-energy x is only 5 dB, a more-realistic number for a speech recognition system. In this case, both the log-normal approximation and the first-order VTS approximation are good estimates of the mean of y. Again, mostly because they ignore the cross-term, neither approximation gives a reliable estimate for the standard deviation of y in the region between −20 dB and 20 dB.

33.5.5 Strategies for Highly Nonstationary Noises

So far, this section has dealt only with stationary noise. That is, it assumed the noise cepstra for a given utterance are well modeled by a Gaussian distribution. In practice, though, there are many nonstationary noises that do not fit that model. What is worse, nonstationary noises can match a random word in the system's lexicon better than the silence model. In this case, the benefit of using speech recognition vanishes quickly.

One solution to the problem of nonstationary noise is to use a more-complex noise model with standard model adaptation algorithms. For instance, the Gaussian noise model can be replaced with a Gaussian mixture model (GMM) or hidden Markov model (HMM).

With a GMM noise model, the decoding becomes more computationally intensive. Before, a Gaussian noise model transformed each component of the clean speech model into one component in the adapted noisy speech model. Now, assume that the noise is independent of speech and is modeled by a mixture of M Gaussian components. Each Gaussian component in the clean speech model will mix independently with every component of the noise GMM. As a result, the acoustic model will grow by a factor of M.

An HMM noise model can provide a much better fit to nonstationary noises [33.40, 41]. However, to efficiently use an HMM noise model, the decoder needs to be modified to perform a three-dimensional Viterbi search which evaluates every possible speech state and noise state at every frame. The computational complexity of performing this *speech/noise decomposition* is very large, though in theory it can handle nonstationary noises quite well.

Alternatively, dedicated whole-word garbage models can bring some of the advantages of an HMM noise model without the additional cost of a three-dimensional Viterbi search. In this technique [33.42], new words are created in the acoustic and language models to cover nonstationary noises such as lip smacks, throat clearings, coughs, and filler words such as *uhm* and *uh*. These nuisance words can be successfully recognized and ignored during non-speech regions, where they tend to cause the most damage.

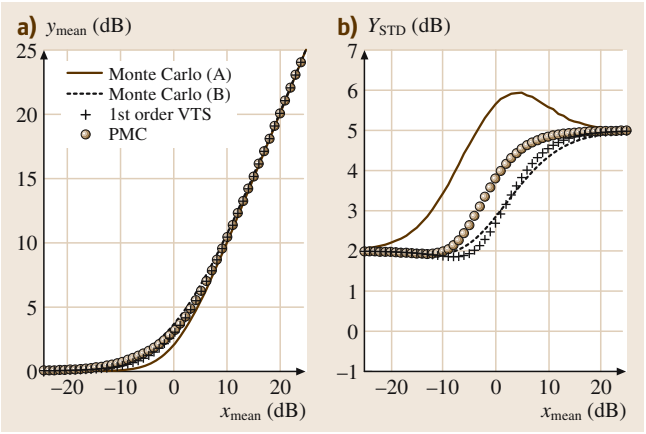


Fig. 33.9a,b Mean and standard deviation of noisy speech y in dB. The distribution of the noise log-spectrum n is Gaussian with mean of 0 dB and standard deviation of 2 dB. The distribution of the clean log-spectrum x is Gaussian with a standard deviation of 5 dB and a mean varying from −25 dB to 25 dB. Both log-normal PMC and the first-order VTS approximation make good estimates compared to a Monte Carlo simulation of (33.27) when the cross-term is ignored **(b)**, although the standard deviation is revealed as an underestimate when the cross-term is included **(a)**

33.6 Structured Feature Enhancement

This section presents several popular methods for structured feature enhancement. Whereas the techniques of Sect. 33.5 adapt the recognizer's acoustic model parameters, the techniques discussed here use mathematical models of the noise corruption to enhance the features before they are presented to the recognizer.

Methods in this class have a rich history. *Boll* [33.43] pioneered the use of spectral subtraction for speech enhancement almost thirty years ago, and it is still found in many systems today. *Ephraim* and *Malah* [33.44] invented their logarithmic minimum mean-squared error short-time spectral-amplitude (logMMSE STSA) estimator shortly afterwards, which forms the basis of many of today's advanced systems. Whereas these earlier approaches use weak speech models, today's most promising systems use stronger models, coupled with vector Taylor series speech enhancement [33.45].

Some of these techniques were developed for speech enhancement for human consumption. Unfortunately, what sounds good to a human can confuse an automatic speech recognition system, and automatic systems can tolerate distortions that are unacceptable to human listeners.

33.6.1 Spectral Subtraction

Spectral subtraction is built on the assumption that the observed power spectrum is approximately the sum of the power spectra for the clean signal and the noise. Although this assumption holds in the expected sense, as we have seen before, in any given frame it is only a rough approximation (Sect. 33.4):

$$|Y(f)|^2 \approx |X(f)|^2 + |N(f)|^2 \,. \qquad (33.63)$$

Using (33.63), the clean power spectrum can be estimated by subtracting an estimate of the noise power spectrum from the noisy power spectrum:

$$|\hat{X}(f)|^2 = |Y(f)|^2 - |\hat{N}(f)|^2 = |Y(f)|^2 H_{\text{ss}}^2(f)\,, \tag{33.64}$$

where the equivalent spectral subtraction filter in the power spectral domain is

$$H_{\text{ss}}^2(f) = 1 - \frac{1}{\text{SNR}(f)}\,, \tag{33.65}$$

and the frequency-dependent signal-to-noise ratio $\text{SNR}(f)$ is

$$\text{SNR}(f) = \frac{|Y(f)|^2}{|\hat{N}(f)|^2}\,. \tag{33.66}$$

The weakest point in many speech enhancement algorithms is the method for estimating the noise power spectrum. More advanced enhancement algorithms can be less sensitive to this estimate, but spectral subtraction is quite sensitive. The easiest option is to assume the noise is stationary, and obtain an estimate using the average periodogram over M frames that are known to be just noise

$$|\hat{N}(f)|^2 = \frac{1}{M}\sum_{i=0}^{M-1} |Y_i(f)|^2\,. \tag{33.67}$$

In practice, there is no guarantee that the spectral subtraction filter in (33.65) is nonnegative, which violates the fundamental nature of power spectra. In fact, it is easy to see that noise frames do not comply. To enforce this constraint, *Boll* [33.43] suggested modifying the filter as

$$H_{\text{ss}}^2(f) = \max\left(1 - \frac{1}{\text{SNR}(f)}, a\right) \tag{33.68}$$

with $a \geq 0$, so that the filter is always positive.

This implementation results in output speech that has significantly less noise, though it exhibits what is called *musical noise* [33.46]. This is caused by frequency bands f for which $|Y(f)|^2 \approx |\hat{N}(f)|^2$. As shown in Fig. 33.7, a frequency f_0 for which $|Y(f_0)|^2 < |\hat{N}(f_0)|^2$ is attenuated by A dB, whereas a neighboring frequency f_1, where $|Y(f_1)|^2 > |\hat{N}(f_1)|^2$, has a much smaller attenuation. These rapid changes of attenuation over frequency introduce tones at varying frequencies that appear and disappear rapidly.

The main reason for the presence of musical noise is that the estimates of $\text{SNR}(f)$ are poor. This is partly because the $\text{SNR}(f)$ are computed independently for every time and frequency, even though it would be more reasonable to assume that nearby values are correlated. One possibility is to smooth the filter in (33.68) over time, frequency, or both. This approach suppresses a smaller amount of noise, but it does not distort the signal as much, and thus may be preferred. An easy way to smooth over time is to mix a small portion of the previous SNR estimate into the current frame:

$$\text{SNR}(f,t) = \gamma\,\text{SNR}(f,t-1) + (1-\gamma)\frac{|Y(f)|^2}{|\hat{N}(f)|^2}\,. \tag{33.69}$$

This smoothing yields more-accurate SNR measurements and thus less distortion, at the expense of reduced noise suppression.

Other enhancements to the basic algorithm have been proposed to reduce the musical noise. Sometimes (33.68) is generalized to

$$f_{\text{ms}}(x) = \left[\max\left(1 - \frac{1}{x^{\frac{\alpha}{2}}}, a\right)\right]^{\frac{1}{\alpha}}\,, \tag{33.70}$$

where $\alpha = 2$ corresponds to the *power spectral subtraction rule* in (33.64), and $\alpha = 1$ corresponds to the *magnitude subtraction rule* (plotted in Fig. 33.7 for $A = 10\,\text{dB}$):

$$g_{\text{ms}}(\overline{x}) = \max\left[20\log_{10}\left(1 - 10^{-\frac{\overline{x}}{5}}\right), -A\right]\,. \tag{33.71}$$

Another variation, called oversubtraction, consists of multiplying the estimate of the noise power spectral density $|\hat{N}(f)|^2$ in (33.67) by a constant $10^{\frac{\beta}{10}}$, where $\beta > 0$, which causes the power spectral subtraction rule to be transformed to another function

$$g_{\text{ms}}(\overline{x}) = \max\left\{10\log_{10}\left[1 - 10^{-\frac{(\overline{x}-\beta)}{10}}\right], -A\right\}\,. \tag{33.72}$$

This causes $|Y(f)|^2 < |\hat{N}(f)|^2$ to occur more often than $|Y(f)|^2 > |\hat{N}(f)|^2$ for frames for which $|Y(f)|^2 \approx |\hat{N}(f)|^2$, and thus reduces the musical noise.

Since the normal cepstral processing for speech recognition includes finding the power spectrum for each frame of data, spectral subtraction can be performed directly in the feature extraction module, without the need for resynthesis. Instead of operating directly on the full-resolution power spectrum, a popular alternative is to use the mel-frequency power spectrum. This does not change the motivation or derivation for spectral subtraction, but the mel-frequency power spectrum is more stable and less susceptible to musical noise.

Because spectral techniques like spectral subtraction can be implemented with very little computational cost, they are popular in embedded applications. In particular, the advanced front-end (AFE) standard produced by the ETSI DSW working group [33.47, 48]

Table 33.10 ETSI advanced front-end word accuracy for Aurora 2. This low-resource front end produces respectable results on the task

Acoustic model	Test set A	Test set B	Test set C	Average
Clean	89.27	87.92	88.53	88.58
Multistyle	93.74	93.26	92.21	93.24

uses a variation of this technique in its noise reduction module. The complete AFE noise reduction process consists of a two-stage time-smoothed frequency-domain filtering [33.49], time-domain noise reduction [33.50], SNR-dependent waveform processing [33.51], and an online variant of CMN [33.52].

Table 33.10 presents the performance of the ETSI AFE on the Aurora 2 clean and multistyle tasks. Despite its low computational cost, the multistyle AFE has only 10% more errors than the best system described in this chapter (Table 33.15). It achieves a average word accuracy of 88.19% when trained with clean data, and 93.24% when trained with multistyle data.

33.6.2 Vector Taylor-Series Speech Enhancement

As shown in Sect. 33.5.3, the VTS approximation can be used to create a noisy speech model from a clean speech model. But, adapting each Gaussian component in a large speech model can be quite computationally expensive.

A lightweight alternative is to leverage a smaller model for speech into a separate enhancement step. The VTS approximation is applied to this smaller model, which is then used to estimate the enhanced features that are sent to an unmodified recognition system. A good comprehensive summary of this approach can be found in [33.45], and the literature is rich with implementations [33.38, 53].

Typically, the clean speech model is a multivariate Gaussian mixture model, and the noise model is a single Gaussian component. The parameters of this prior model include the state-conditional means and variances, $\mu_s^{\boldsymbol{x}}$, $\sigma_s^{\boldsymbol{x}}$, $\mu^{\boldsymbol{n}}$ and $\sigma^{\boldsymbol{n}}$, as well as the mixture component weights $p(s)$:

$$p(\boldsymbol{x}) = \sum_s N\left(\boldsymbol{x}; \mu_s^{\boldsymbol{x}}, \sigma_s^{\boldsymbol{x}}\right) p(s) \,, \tag{33.73}$$

$$p(\boldsymbol{n}) = N(\boldsymbol{n}; \mu^{\boldsymbol{n}}, \sigma^{\boldsymbol{n}}) \,. \tag{33.74}$$

These models are tied together by the nonlinear mixing represented by (33.36), recast as a probability distribution:

$$p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{n}) \approx N[\boldsymbol{y}; \boldsymbol{x} + \boldsymbol{g}(\boldsymbol{x} - \boldsymbol{n}), \Psi^2] \,. \tag{33.75}$$

Here, the variance Ψ^2 chiefly represents the error incurred in ignoring the cross-term produced by mixing the speech and noise (Sect. 33.4), and is in general a function of $(\boldsymbol{x} - \boldsymbol{n})$. For an overview of three reasonable approximations for Ψ^2 [33.38].

This Chapter uses the approximation $\Psi^2 = 0$, which is equivalent to ignoring the effects of the cross-term entirely. We call this approximation the zero-variance model (ZVM). This yields a good fit to the data at the extreme SNR regions, and a slight mismatch in the $\boldsymbol{x} \approx \boldsymbol{n}$ region.

If the VTS techniques from Sect. 33.5.3 were applied directly to the variables $\boldsymbol{x}$ and $\boldsymbol{n}$ for VTS enhancement, the result would be quite unstable [33.38]. Instead, the problem is reformulated in terms of $\boldsymbol{r}$, the instantaneous signal-to-noise ratio, defined as

$$\boldsymbol{r} = \boldsymbol{x} - \boldsymbol{n} \,.$$

By performing VTS on the new variable $\boldsymbol{r}$, the stability problems are circumvented. In the end, an estimate for the instantaneous SNR can be mapped back into estimates of $\boldsymbol{x}$ and $\boldsymbol{n}$ through

$$\boldsymbol{x} = \boldsymbol{y} - \boldsymbol{g}(\boldsymbol{r}) \,, \tag{33.76}$$

$$\boldsymbol{n} = \boldsymbol{y} - \boldsymbol{g}(\boldsymbol{r}) - \boldsymbol{r} \,. \tag{33.77}$$

These formulas satisfy the intuition that as the SNR $\boldsymbol{r}$ becomes more positive, $\boldsymbol{x}$ approaches $\boldsymbol{y}$ from below. As the SNR $\boldsymbol{r}$ becomes more negative, $\boldsymbol{n}$ approaches $\boldsymbol{y}$ from below.

The joint PDF for the ZVM is a distribution over the clean speech $\boldsymbol{x}$, the noise $\boldsymbol{n}$, the observation $\boldsymbol{y}$, the SNR $\boldsymbol{r}$, and the speech state s:

$$p(\boldsymbol{y}, \boldsymbol{r}, \boldsymbol{x}, \boldsymbol{n}, s) = p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{n}) p(\boldsymbol{r}|\boldsymbol{x}, \boldsymbol{n}) p(\boldsymbol{x}, s) p(\boldsymbol{n}) \,.$$

The observation and SNR are both deterministic functions of $\boldsymbol{x}$ and $\boldsymbol{n}$. As a result, the conditional probabilities $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{n})$ and $p(\boldsymbol{r}|\boldsymbol{x}, \boldsymbol{n})$ can be represented by Dirac delta functions:

$$p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{n}) = \delta[\ln(\mathrm{e}^{\boldsymbol{x}} + \mathrm{e}^{\boldsymbol{n}}) - \boldsymbol{y}] \,, \tag{33.78}$$

$$p(\boldsymbol{r}|\boldsymbol{x}, \boldsymbol{n}) = \delta(\boldsymbol{x} - \boldsymbol{n} - \boldsymbol{r}) \,. \tag{33.79}$$

Table 33.11 Word accuracy for Aurora 2, using VTS speech enhancement and an acoustic model trained on clean data. Enhancement is performed on static cepstra, which are then used to compute the dynamic coefficients used in recognition. This enhancement technique does better than any of the normalization techniques

Acoustic model	Iterations	Set A	Set B	Set C	Average
Clean	0	88.59	88.06	86.92	88.04
Clean	1	89.67	89.05	87.37	88.96
Clean	2	89.92	89.39	87.28	89.18
Clean	3	89.99	89.48	87.05	89.20

Table 33.12 Word accuracy for Aurora 2, using VTS speech enhancement and an acoustic model trained on enhanced multistyle data. As in Table 33.11, the dynamic coefficients are calculated from enhanced static coefficients. In the end, the result is not as good as simple feature normalization alone on a multistyle acoustic model

Acoustic model	Iterations	Set A	Set B	Set C	Average
Multistyle	0	92.58	91.14	92.52	91.99
Multistyle	1	92.79	91.27	92.24	92.07
Multistyle	2	92.86	91.35	92.10	92.11
Multistyle	3	92.82	91.35	91.94	92.06

This allows us to marginalize the continuous variables $\boldsymbol{x}$ and $\boldsymbol{n}$, as in

$$\begin{aligned} p(\boldsymbol{y},\boldsymbol{r},s) &= \int \mathrm{d}\boldsymbol{x} \int \mathrm{d}\boldsymbol{n}\, p(\boldsymbol{y},\boldsymbol{r},\boldsymbol{x},\boldsymbol{n},s) \\ &= N[\boldsymbol{y}-\boldsymbol{g}(\boldsymbol{r});\mu_s^{\boldsymbol{x}},\sigma_s^{\boldsymbol{x}}]p(s) \\ &\quad \times N[\boldsymbol{y}-\boldsymbol{g}(\boldsymbol{r})-\boldsymbol{r};\mu^{\boldsymbol{n}},\sigma^{\boldsymbol{n}}]\,. \end{aligned} \tag{33.80}$$

After marginalization, the only remaining continuous hidden variable is $\boldsymbol{r}$, the instantaneous SNR. The behavior of this joint PDF is intuitive. At high SNR,

$$p(\boldsymbol{y},\boldsymbol{r},s) \approx N(\boldsymbol{y};\mu_s^{\boldsymbol{x}},\sigma_s^{\boldsymbol{x}})p(s)N(\boldsymbol{y}-\boldsymbol{r};\mu^{\boldsymbol{n}},\sigma^{\boldsymbol{n}})\,.$$

That is, the observation is modeled as clean speech, and the noise is at a level $\boldsymbol{r}$ units below the observation. The converse is true for low SNR:

$$p(\boldsymbol{y},\boldsymbol{r},s) \approx N(\boldsymbol{y}-\boldsymbol{r};\mu_s^{\boldsymbol{x}},\sigma_s^{\boldsymbol{x}})p(s)N(\boldsymbol{y};\mu^{\boldsymbol{n}},\sigma^{\boldsymbol{n}})\,.$$

To solve the MMSE estimation problem, the nonlinear function $\boldsymbol{g}(\boldsymbol{r})$ in (33.80) is replaced by its Taylor-series approximation, as in Sect. 33.5.3. For now, the expansion point is the expected a priori mean of $\boldsymbol{r}$:

$$\boldsymbol{r}_s^0 = E\{\boldsymbol{r}|s\} = E\{\boldsymbol{x}-\boldsymbol{n}|s\} = \mu_s^{\boldsymbol{x}}-\mu^{\boldsymbol{n}}\,.$$

Using (33.80) and the VTS approximation, it can be shown that $p(\boldsymbol{y}|s)$ has the same form derived in Sect. 33.5.3. Essentially, we perform VTS adaptation of the clean speech GMM to produce a GMM for noisy speech:

$$p(\boldsymbol{y}|s) = N(\boldsymbol{y};\mu_{\boldsymbol{y}|s},\sigma_{\boldsymbol{y}|s})\,, \tag{33.81}$$

$$\mu_{\boldsymbol{y}|s} = \mu_s^{\boldsymbol{x}} + \boldsymbol{g}\left(-\boldsymbol{r}_s^0\right) + \mathbf{G}_s\left(\mu_s^{\boldsymbol{x}}-\mu_s^{\boldsymbol{n}}-\boldsymbol{r}_s^0\right)\,, \tag{33.82}$$

$$\sigma_{\boldsymbol{y}|s} = \mathbf{G}_s\sigma_s^{\boldsymbol{x}}\mathbf{G}_s^{\mathrm{T}} + (\mathbf{I}-\mathbf{G}_s)\sigma^{\boldsymbol{n}}(\mathbf{I}-\mathbf{G}_s)^{\mathrm{T}}\,. \tag{33.83}$$

When using the recommended expansion point, (33.82) simplifies to

$$\mu_{\boldsymbol{y}|s} = \mu^{\boldsymbol{x}} + \boldsymbol{g}\left(\mu^{\boldsymbol{n}}-\mu_s^{\boldsymbol{x}}\right)\,.$$

It is also straightforward to derive the conditional posterior $p(\boldsymbol{r}|\boldsymbol{y},s)$, as in (33.84). As in the other iterative VTS algorithms, we can use the expected value $E[\boldsymbol{r}|\boldsymbol{y},s] = \mu_s^{\boldsymbol{r}}$ as a new expansion point for the vector Taylor-series parameters and iterate.

$$\begin{aligned} p(\boldsymbol{r}|\boldsymbol{y},s) &= N\left(\boldsymbol{r};\mu_s^{\boldsymbol{r}},\sigma_s^{\boldsymbol{r}}\right)\,, \\ \left(\sigma_s^{\boldsymbol{r}}\right)^{-1} &= (\mathbf{I}-\mathbf{G}_s)^{\mathrm{T}}\left(\sigma_s^{\boldsymbol{x}}\right)^{-1}(\mathbf{I}-\mathbf{G}_s) \\ &\quad + (\mathbf{G}_s)^{\mathrm{T}}(\sigma^{\boldsymbol{n}})^{-1}\mathbf{G}_s\,, \\ \mu_s^{\boldsymbol{r}} &= \mu_s^{\boldsymbol{x}} - \mu^{\boldsymbol{n}} + \sigma_s^{\boldsymbol{r}}\left[(\mathbf{G}_s-\mathbf{I})^{\mathrm{T}}\left(\sigma_s^{\boldsymbol{x}}\right)^{-1}\right. \\ &\quad \left. + (\mathbf{G}_s)^{\mathrm{T}}(\sigma^{\boldsymbol{n}})^{-1}\right](\boldsymbol{y}-\mu_{\boldsymbol{y}|s})\,. \end{aligned} \tag{33.84}$$

After convergence, we compute an estimate of x from the parameters of the approximate model:

$$\hat{\boldsymbol{x}} = \sum_s E[\boldsymbol{x}|\boldsymbol{y},s]p(s|\boldsymbol{y})\,,$$

$$E[\boldsymbol{x}|\boldsymbol{y},s] \approx \boldsymbol{y} - \ln(\mathrm{e}^{\mu_s^{\boldsymbol{r}}}+1) + \mu_s^{\boldsymbol{r}}\,.$$

Here, (33.76) has been used to map $E[\boldsymbol{r}|\boldsymbol{y},s] = \mu_s^{\boldsymbol{r}}$ to $E[\boldsymbol{x}|\boldsymbol{y},s]$. Since the transformation is nonlinear, our estimate for $\hat{\boldsymbol{x}}$ is not the optimal MMSE estimator.

Tables 33.11–33.14 evaluate accuracy on Aurora 2 for several different VTS speech enhancement configurations. For all experiments, the clean speech GMM consisted of 32 diagonal components trained on the clean Aurora 2 training data.

Table 33.13 Word accuracy for Aurora 2, using VTS speech enhancement and an acoustic model trained on clean data. Unlike Table 33.11, the static cepstral coefficients are enhanced and then concatenated with noisy dynamic coefficients. The end result is worse than computing the dynamic coefficients from enhanced static coefficients

Acoustic model	Iterations	Set A	Set B	Set C	Average
Clean	0	83.56	84.60	82.52	83.77
Clean	1	84.38	85.48	82.82	84.51
Clean	2	84.60	85.72	82.76	84.68
Clean	3	84.68	85.81	82.65	84.73

Table 33.14 Word accuracy for Aurora 2, using VTS speech enhancement and an acoustic model trained on enhanced multistyle data. Unlike Table 33.12, the static cepstral coefficients are enhanced and then concatenated with noisy dynamic coefficients. The end result is better than computing the dynamic coefficients from enhanced static coefficients

Acoustic model	Iterations	Set A	Set B	Set C	Average
Multistyle	0	93.64	93.05	93.05	93.29
Multistyle	1	93.76	93.14	92.81	93.32
Multistyle	2	93.72	93.19	92.79	93.32
Multistyle	3	93.57	92.95	92.70	93.15

Table 33.15 Word accuracy for Aurora 2, adding CMVN after the VTS speech enhancement of Table 33.14. The result is a very high recognition accuracy

Acoustic model	Normalization	Set A	Set B	Set C	Average
Multistyle	None	93.72	93.19	92.79	93.32
Multistyle	CMVN	94.09	93.46	94.01	93.83

Of course, using the multistyle training data produces better accuracy. Comparing Tables 33.11 and 33.12, it is apparent that multistyle data should be used whenever possible. Note that the result with the clean acoustic model is better than all of the feature normalization techniques explored in this chapter, and better than the VTS adaptation result of Table 33.9. Because VTS enhancement is fast enough to compute new parameters specific to each utterance, it is able to better adapt to the changing conditions within each noise type.

Although the speech recognition features consist of static and dynamic cepstra, the VTS enhancement is only defined on the static cepstra. As a result, there are two options for computing the dynamic cepstra. In Tables 33.11 and 33.12, the dynamic cepstra were computed from the enhanced static cepstra. In the corresponding Tables 33.13 and 33.14, the dynamic cepstra were computed from the noisy static cepstra. In the former case, the entire feature vector is affected by the enhancement algorithm, and in the latter, only the static cepstra are modified.

Using the noisy dynamic cepstra turns out to be better for the multistyle acoustic model, but worse for the clean acoustic model. Under the clean acoustic model, the benefit of the enhancement outweighs the distortion it introduces. However, the multistyle acoustic model is already able to learn and generalize the dynamic coefficients from noisy speech. Table 33.14 shows that the best strategy is to only enhance the static coefficients.

Finally, consider the effect of adding feature normalization to the system. Normalization occurs just after the full static and dynamic feature vector is created, and before it is used by the recognizer. Table 33.15 presents the final accuracy achieved by adding CMVN to the result of Table 33.14. Even though the accuracy of the multistyle model was already quite good, CMVN reduces the error rate by another 7.6% relative.

33.7 Unifying Model and Feature Techniques

The front- and back-end methods discussed so far can be mixed and matched to good performance. This section introduces two techniques that achieve better accuracy through a tighter in-

tegration of the front- and back-end robustness techniques.

33.7.1 Noise Adaptive Training

Section 33.2 discussed how the recognizer's HMMs can be adapted to a new acoustical environment. Section 33.6 dealt with cleaning the noisy feature without retraining the HMMs. It is logical to consider a combination of both, where the features are cleaned to remove noise and channel effects and then the HMMs are retrained to take into account that this processing stage is not perfect.

It was shown in [33.19] that a combination of feature enhancement and matched condition training can achieve a lower word error rate than feature enhancement alone. This paper demonstrated how, by introducing a variant of the enhancement algorithm from Sect. 33.3.4, very low error rates could be achieved.

These low error rates are hard to obtain in practice, because they assume preknowledge of the exact noise type and level, which in general is difficult to obtain. On the other hand, this technique can be effectively combined with the multistyle training discussed in Sect. 33.2.1.

33.7.2 Uncertainty Decoding and Missing Feature Techniques

Traditionally, the speech enhancement algorithms from Sect. 33.6 output an estimate of the clean speech to be used by the speech recognition system. However, the accuracy of the noise removal process can vary from frame to frame and from dimension to dimension in the feature stream.

Uncertainty decoding [33.54] is a technique where the feature enhancement algorithm associates a confidence with each value that it outputs. In frames with a high signal-to-noise ratio, the enhancement can be very accurate and would be associated with high confidence. Other frames, where some or all of the speech has been buried in noise, would have low confidence.

The technique is implemented at the heart of the speech recognition engine, where many Gaussian mixture components are evaluated. When recognizing uncorrupted speech cepstra, the purpose of these evaluations is to discover the probability of each clean observation vector, conditioned on the mixture index, $p_{\boldsymbol{x}|m}(\boldsymbol{x}|m)$, for each Gaussian component in the speech model used by the recognizer.

If the training and testing conditions do not match, as is the case in noise-corrupted speech recognition, one option is to ignore the imperfections of the noise removal, and evaluate $p_{\boldsymbol{x}|m}(\hat{\boldsymbol{x}}|m)$. This is the classic case of passing the output of the noise removal algorithm directly to the recognizer.

With the uncertainty decoding technique, the joint conditional PDF $p(\boldsymbol{x}, \boldsymbol{y}|m)$ is generated and marginalized over all possible unseen clean-speech cepstra:

$$p(\boldsymbol{y}|m) = \int_{-\infty}^{\infty} p(\boldsymbol{y}, \boldsymbol{x}|m)\,\mathrm{d}\boldsymbol{x}\,. \tag{33.85}$$

Under this framework, instead of just providing cleaned cepstra, the speech enhancement process also estimates the conditional distribution $p(\boldsymbol{y}|\boldsymbol{x}, m)$, as a function of $\boldsymbol{x}$. For ease of implementation, it is generally assumed that $p(\boldsymbol{y}|\boldsymbol{x})$ is independent of m:

$$p(\boldsymbol{y}|\boldsymbol{x}, m) \approx p(\boldsymbol{y}|\boldsymbol{x}) = \alpha N\big(\boldsymbol{x}; \hat{\boldsymbol{x}}(\boldsymbol{y}), \sigma^2_{\hat{\boldsymbol{x}}}(\boldsymbol{y})\big)\,, \tag{33.86}$$

where α is independent of $\boldsymbol{x}$, and therefore can be ignored by the decoder. Note that $\hat{\boldsymbol{x}}$ and $\sigma^2_{\hat{\boldsymbol{x}}}$ are always functions of $\boldsymbol{y}$; the cumbersome notation is dropped for the remainder of this discussion.

Finally, the probability for the observation $\boldsymbol{y}$, conditioned on each acoustic model Gaussian mixture component m, can be calculated:

$$\begin{aligned} p(\boldsymbol{y}|m) &= \int_{-\infty}^{\infty} p(\boldsymbol{y}|\boldsymbol{x}, m)p(\boldsymbol{x}|m)\,\mathrm{d}\boldsymbol{x} \\ &\propto \int_{-\infty}^{\infty} N\big(\hat{\boldsymbol{x}}; \boldsymbol{x}, \boldsymbol{\sigma}^2_{\hat{\boldsymbol{x}}}\big) N\big(\boldsymbol{x}; \boldsymbol{\mu}_m, \boldsymbol{\sigma}^2_m\big)\,\mathrm{d}\boldsymbol{x} \\ &= N\big(\hat{\boldsymbol{x}}; \boldsymbol{\mu}_m, \boldsymbol{\sigma}^2_m + \boldsymbol{\sigma}^2_{\hat{\boldsymbol{x}}}\big)\,. \end{aligned} \tag{33.87}$$

This formula is evaluated for each Gaussian mixture component in the decoder, $p(\boldsymbol{x}|m) = N(\boldsymbol{x}, \boldsymbol{\mu}_m, \boldsymbol{\sigma}^2_m)$.

As can be observed in (33.87), the uncertainty output from the front end increases the variance of the Gaussian mixture component, producing an effective smoothing in cases where the front end is uncertain of the true value of the cleaned cepstra.

Two special cases exist for uncertainty decoding. In the absence of uncertainty information from the noise removal process, we can either assume that there is no uncertainty or that there is complete uncertainty.

If there were no uncertainty, then $\boldsymbol{\sigma}^2_{\hat{\boldsymbol{x}}} = 0$. The probability of the observation $\boldsymbol{y}$ for each acoustic model Gaussian mixture component m simplifies to:

$$p(\boldsymbol{y}|m) = p(\hat{\boldsymbol{x}}|m) = N\big(\hat{\boldsymbol{x}}; \boldsymbol{\mu}_m, \boldsymbol{\sigma}^2_m\big)\,. \tag{33.88}$$

This is the traditional method of passing features directly from the noise removal algorithm to the decoder.

If there were complete uncertainty of any of the cepstral coefficients, the corresponding $\sigma_{\hat{x}}^2$ would approach infinity. That coefficient would have no effect on the calculation of $p(\boldsymbol{y}|m)$. This is desirable behavior, under the assumption that the coefficient could not contribute to discrimination.

Both of these extreme cases are similar to the computations performed when using hard thresholds with missing-feature techniques [33.55]. There has been some success in incorporating heuristic soft thresholds with missing-feature techniques [33.56], but without the benefits of a rigorous probabilistic framework.

33.8 Conclusion

To prove a newly developed system, it can be tested on any one of a number of standard noise-robust speech recognition tasks. Because a great number of researchers are publishing systematic results on the same tasks, the relative value of complete solutions can be easily assessed.

When building a noise-robust speech recognition system, there exist several simple techniques that should be tried before more-complex strategies are invoked. These include the feature normalization techniques of Sect. 33.3, as well as the model retraining methods of Sect. 33.2.

If state-of-the-art performance is required with a small amount of adaptation data, then the structured techniques of Sects. 33.5 and 33.6 can be implemented. Structured model adaptation carries with it an expensive computational burden, whereas structured feature enhancement is more lightweight but less accurate.

Finally, good compromises between the accuracy of model adaptation and the speed of feature enhancement can be achieved through a tighter integration of the front and back end, as shown in Sect. 33.7.

References

33.1 H.G. Hirsch, D. Pearce: The AURORA experimental framework for the performance evaluations of speech recognition systems under noisy conditions, ISCA ITRW ASR2000 "Automatic Speech Recognition: Challenges for the Next Millennium" (2000)

33.2 R.G. Leonard, G. Doddington: *Tidigits* (Linguistic Data Consortium, Philadelphia 1993)

33.3 D. Pierce, A. Gunawardana: Aurora 2.0 speech recognition in noise: Update 2. Complex backend definition for Aurora 2.0, http://icslp2002.colorado.edu/special_sessions/aurora (2002)

33.4 A. Moreno, B. Lindberg, C. Draxler, G. Richard, K. Choukri, J. Allen, S. Euler: SpeechDat-Car: A large speech database for automotive environments, Proc. 2nd Int. Conf. Language Resources and Evaluation (2000)

33.5 J. Garofalo, D. Graff, D. Paul, D. Pallett: *CSR-I (WSJ0) Complete* (Linguistic Data Consortium, Philadelphia 1993)

33.6 A. Varga, H.J.M. Steeneken, M. Tomlinson, D. Jones: The NOISEX-92 study on the effect of additive noise on automatic speech recognition. Tech. Rep. Defence Evaluation and Research Agency (DERA) (Speech Research Unit, Malvern 1992)

33.7 A. Schmidt-Nielsen: *Speech in Noisy Environments (SPINE) Evaluation Audio* (Linguistic Data Consortium, Philadelphia 2000)

33.8 R.P. Lippmann, E.A. Martin, D.P. Paul: Multi-style training for robust isolated-word speech recognition, Proc. IEEE ICASSP (1987) pp. 709–712

33.9 M. Matassoni, M. Omologo, D. Giuliani: Hands-free speech recognition using a filtered clean corpus and incremental HMM adaptation, Proc. IEEE ICASSP (2000) pp. 1407–1410

33.10 G. Saon, J.M. Huerta, E.-E. Jan: Robust digit recognition in noisy environments: The Aurora 2 system, Proc. Eurospeech **2001** (2001)

33.11 C.-P. Chen, K. Filali, J.A. Bilmes: Frontend post-processing and backend model enhancement on the Aurora 2.0/3.0 databases, Int. Conf. Spoken Language Process. (2002)

33.12 B.S. Atal: Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification, J. Acoust. Soc. Am. **55**(6), 1304–1312 (1974)

33.13 B.W. Gillespie, L.E. Atlas: Acoustic diversity for improved speech recognition in reverberant environments, Proc. IEEE ICASSP **I**, 557–560 (2002)

33.14 A. de la Torre, A.M. Peinado, J.C. Segura, J.L. Perez-Cordoba, M.C. Benítez, A.J. Rubio: Histogram equalization of speech representation for robust speech recognition, IEEE Trans. Speech Audio Process. **13**(3), 355–366 (2005)

33.15 M.G. Rahim, B.H. Juang: Signal bias removal by maximum likelihood estimation for robust telephone speech recognition, IEEE Trans. Speech Audio Process. **4**(1), 19–30 (1996)

33.16 A. Acero, X.D. Huang: Augmented cepstral normalization for robust speech recognition, Proc. IEEE Workshop on Automatic Speech Recognition (1995)

33.17 J. Ramírez, J.C. Segura, C. Benítez, L. García, A. Rubio: Statistical voice activity detection using a multiple observation likelihood ratio test, IEEE Signal Proc. Lett. **12**(10), 689–692 (2005)

33.18 H. Hermansky, N. Morgan: RASTA processing of speech, IEEE Trans. Speech Audio Process. **2**(4), 578–589 (1994)

33.19 L. Deng, A. Acero, M. Plumpe, X.D. Huang: Large-vocabulary speech recognition under adverse acoustic environments, Int. Conf. Spoken Language Process. (2000)

33.20 A. Acero: *Acoustical and Environmental Robustness in Automatic Speech Recognition* (Kluwer Academic, Boston 1993)

33.21 P. Moreno: *Speech Recognition in Noisy Environments*, Ph.D. Thesis (Carnegie Mellon University, Pittsburgh 1996)

33.22 A. Acero, R.M. Stern: Environmental robustness in automatic speech recognition, Proc. IEEE ICASSP (1990) pp. 849–852

33.23 J. Droppo, A. Acero: Maximum mutual information SPLICE transform for seen and unseen conditions, Proc. Interspeech Conf. (2005)

33.24 J. Wu, Q. Huo: An environment compensated minimum classification error training approach and its evaluation on Aurora 2 database, Proc. ICSLP **1**, 453–456 (2002)

33.25 D. Povey, B. Kingsbury, L. Mangu, G. Saon, H. Soltau, G. Zweig: fMPE: Discriminatively trained features for speech recognition, Proc. IEEE ICASSP (2005)

33.26 S. Tamura, A. Waibel: Noise reduction using connectionist models, Proc. IEEE ICASSP (1988) pp. 553–556

33.27 L. Neumeyer, M. Weintraub: Probabilistic optimum filtering for robust speech recognition, Proc. IEEE ICASSP **1**, 417–420 (1994)

33.28 A. Acero, R.M. Stern: Robust speech recognition by normalization of the acoustic space, Proc. IEEE ICASSP **2**, 893–896 (1991)

33.29 M.J. Gales: *Model Based Techniques for Noise Robust Speech Recognition*, Ph.D. Thesis (Cambridge University, Cambridge 1995)

33.30 E.A. Wan, R.V.D. Merwe, A.T. Nelson: Dual estimation and the unscented transformation. In: *Advances in Neural Information Processing Systems*, ed. by S.A. Solla, T.K. Leen, K.R. Muller (MIT Press, Cambridge 2000) pp. 666–672

33.31 P.J. Moreno, B. Raj, R.M. Stern: A vector taylor series approach for environment independent speech recognition, Proc. IEEE ICASSP (1996) pp. 733–736

33.32 B.J. Frey, L. Deng, A. Acero, T. Kristjansson: ALGONQUIN: Iterating Laplace's method to remove multiple types of acoustic distortion for robust speech recognition, Proc. Eurospeech (2001)

33.33 J. Droppo, A. Acero, L. Deng: A nonlinear observation model for removing noise from corrupted speech log mel-spectral energies, Proc. Int. Conf. Spoken Language Process. (2002)

33.34 C. Couvreur, H. Van Hamme: Model-based feature enhancement for noisy speech recognition, Proc. IEEE ICASSP **3**, 1719–1722 (2000)

33.35 J. Droppo, A. Acero: Noise robust speech recognition with a switching linear dynamic model, Proc. IEEE ICASSP (2004)

33.36 B. Raj, R. Singh, R. Stern: On tracking noise with linear dynamical system models, Proc. IEEE ICASSP **1**, 965–968 (2004)

33.37 H. Shimodaira, N. Sakai, M. Nakai, S. Sagayama: Jacobian joint adaptation to noise, channel and vocal tract length, Proc. IEEE ICASSP **1**, 197–200 (2002)

33.38 J. Droppo, L. Deng, A. Acero: A comparison of three non-linear observation models for noisy speech features, Proc. Eurospeech Conf. (2003)

33.39 R.A. Gopinath, M.J.F. Gales, P.S. Gopalakrishnan, S. Balakrishnan-Aiyer, M.A. Picheny: Robust speech recognition in noise – performance of the IBM continuous speech recognizer on the ARPA noise spoke task, Proc. ARPA Workshop on Spoken Language Systems Technology (1995) pp. 127–133

33.40 A.P. Varga, R.K. Moore: Hidden markov model decomposition of speech and noise, Proc. IEEE ICASSP (1990) pp. 845–848

33.41 A. Acero, L. Deng, T. Kristjansson, J. Zhang: HMM adaptation using vector taylor series for noisy speech recognition, Int. Conf. Spoken Language Processing (2000)

33.42 W. Ward: Modeling non-verbal sounds for speech recognition, Proc. Speech and Natural Language Workshop (1989) pp. 311–318

33.43 S.F. Boll: Suppression of acoustic noise in speech using spectral subtraction, IEEE T. Acoust. Speech **24**(April), 113–120 (1979)

33.44 Y. Ephraim, D. Malah: Speech enhancement using a minimum mean-square error log-spectral amplitude estimator, IEEE Trans. Acoust. Speech Signal Process., Vol. ASSP-33 (1985) pp. 443–445

33.45 V. Stouten: *Robust Automatic Speech Recognition in Time-varying Environments*, Ph.D. Thesis (K. U. Leuven, Leuven 2006)

33.46 M. Berouti, R. Schwartz, J. Makhoul: Enhancement of speech corrupted by acoustic noise, Proc. IEEE ICASSP (1979) pp. 208–211

33.47 ETSI ES 2002 050 Recommendation: Speech processing, transmission and quality aspects (STQ);

distributed speech recognition; advanced front-end feature extraction algorithm (2002)
33.48 D. Macho, L. Mauuary, B. Noê, Y.M. Cheng, D. Ealey, D. Jouvet, H. Kelleher, D. Pearce, F. Saadoun: Evaluation of a noise-robust DSR front-end on Aurora databases, Proc. ICSLP (2002) pp. 17–20
33.49 A. Agarwal, Y.M. Cheng: Two-stage mel-warped Wiener filter for robust speech recognition, Proc. ASRU (1999)
33.50 B. Noê, J. Sienel, D. Jouvet, L. Mauuary, L. Boves, J. de Veth, F. de Wet: Noise reduction for noise robust feature extraction for distributed speech recognition, Proc. Eurospeech (2001) pp. 201–204
33.51 D. Macho, Y.M. Cheng: SNR-Dependent waveform processing for improving the robustness of ASR front-end, Proc. IEEE ICASSP (2001) pp. 305–308
33.52 L. Mauuary: Blind equalization in the cepstral domain for robust telephone based speech recognition, Proc. EUSPICO **1**, 359–363 (1998)
33.53 M. Afify, O. Siohan: Sequential estimation with optimal forgetting for robust speech recognition, IEEE Trans. Speech Audio Process. **12**(1), 19–26 (2004)
33.54 J. Droppo, A. Acero, L. Deng: Uncertainty decoding with SPLICE for noise robust speech recognition, Proc. IEEE ICASSP (2002)
33.55 M. Cooke, P. Green, L. Josifovski, A. Vizinho: Robust automatic speech recognition with missing and unreliable acoustic data, Speech Commun. **34**(3), 267–285 (2001)
33.56 J.P. Barker, M. Cooke, P. Green: Robust ASR based on clean speech models: An evaluation of missing data techniques for connected digit recognition in noise, Proc. Eurospeech **2001**, 213–216 (2001)

26. Historical Perspective of the Field of ASR/NLU

L. Rabiner, B.-H. Juang

The quest for a machine that can recognize and understand speech, from any speaker, and in any environment has been the holy grail of speech recognition research for more than 70 years. Although we have made great progress in understanding how speech is produced and analyzed, and although we have made enough advances to build and deploy in the field a number of viable speech recognition systems, we still remain far from the ultimate goal of a machine that communicates naturally with any human being. It is the goal of this section to document the history of research in speech recognition and natural language understanding, and to point out areas where great progress has been made, along with the challenges that remain to be solved in the future.

26.1 ASR Methodologies

Speech recognition research has been on-going for more than 70 years. Over that period there have been at least four generations of approaches, and we forecast a fifth generation that is being formulated based on current research themes. The five generations, and the technology themes associated with each of them, are as follows.

1. Generation 1 (1930s to 1950s): use of ad hoc methods to recognize sounds, or small vocabularies of isolated words.
2. Generation 2 (1950s to 1960s): use of acoustic–phonetic approaches to recognize phonemes, phones, or digit vocabularies.
3. Generation 3 (1960s to 1980s): use of pattern recognition approaches to speech recognition of small to medium-sized vocabularies of isolated and connected word sequences, including use of linear predictive coding (LPC) as the basic method of spectral analysis; use of LPC distance measures for pattern similarity scores; use of dynamic programming methods for time aligning patterns; use of pattern recognition methods for clustering multiple patterns into consistent reference patterns; use of vector quantization (VQ) codebook methods for data reduction and reduced computation.
4. Generation 4 (1980s to 2000s): use of hidden Markov model (HMM) statistical methods for modeling speech dynamics and statistics in a continuous speech recognition system; use of forward-backward and segmental K-means training methods; use of Viterbi alignment methods; use of maximum likelihood (ML) and various other performance criteria and methods for optimizing statistical models; introduction of neural network (NN) methods for estimating conditional probability densities; use of adaptation methods that modify the parameters associated with either the speech signal or the statistical model so as to enhance the compatibility between model and data for increased recognition accuracy.

5. Generation 5 (2000s to 2020s): use of parallel processing methods to increase recognition decision reliability; combinations of HMMs and acoustic–phonetic approaches to detect and correct linguistic irregularities; increased robustness for recognition of speech in noise; machine learning of optimal combinations of models.

The generations are not distinct, as most of the key ideas that define each generation were formulated in earlier generations. However, the indicated time periods for each generation represent the eras in which most of the research took place and the times when the resulting technology became the standard for most recognition systems of that era.

We discuss a range of technology issues and the way they have evolved in each of the speech recognition generations in the remainder of this paper.

In the area of natural language understanding (NLU) we have seen three highly overlapping generations of technology evolve including the following:

1. Generation 1: realize task constraints using simple nodal grammars independent of the acoustic realization of the spoken sentence and without regard to disfluencies in speaking.
2. Generation 2: incorporate statistical grammar and utilize finite state networks (FSN) to model acoustics, syntax, and semantics jointly in a single integrated model which can be configured to account for nongrammatical acoustic events, and which can be optimally searched to find the best path corresponding to the most likely spoken sentence consistent with the task grammar and semantics (and even possibly with task pragmatics).
3. Generation 3: utilize more-complex syntax and semantic representations that maximize language coverage and minimize language over-coverage.

Unfortunately, due to space limitations, we will not be able to talk about the technology evolution in the three generations of NLU research; however, Chap. 31 by *Roukos* will cover this area in great detail.

26.1.1 Issues in Speech Recognition

As we examine the progress made in implementing speech recognition and natural language understanding systems over the years, we will see that there are a number of issues that need to be addressed in order to define the operating range of each speech recognition system that is built. These issues include the following:

- Speech unit for recognition: ranging from words down to syllables and finally to phonemes or even phones. Early systems investigated all these types of units with the goal of understanding their robustness to context, speakers and speaking environments
- Vocabulary size: ranging from small (order of 2–100 words), medium (order of 100–1000) words, and large (anything above 1000 words up to unlimited vocabularies). Early systems tackled primarily small-vocabulary recognition problems; modern speech recognizers are all large-vocabulary systems
- Task syntax: ranging from simple tasks with almost no syntax (every word in the vocabulary can follow every other word) to highly complex tasks where the words follow a statistical n-gram language model
- Task perplexity (the average word branching factor): ranging from low values (for simple tasks) to values on the order of 100 for complex tasks whose perplexity approaches that of natural language task
- Speaking mode: ranging from isolated words (or short phrases), to connected word systems (e.g., sequences of digits that form identification codes or telephone numbers), to continuous speech (including both read passages and spontaneous conversational speech)
- Speaker mode: ranging from speaker-trained systems (works only on individual speakers who trained the system) to speaker-adaptive systems (works only after a period of adaptation to the individual speaker's voice) to speaker-independent systems, which can be used by anyone without any additional training. Most modern ASR (automatic speech recognition) systems are speaker independent and are utilized in a range of telecommunication applications. However, for dictation purposes, most systems are still largely speaker dependent and adapt over time to each individual speaker
- Speaking situation: ranging from human-to-machine dialogues to human-to-human dialogues (e.g., as might be needed for language translation systems)
- Speaking environment: ranging from a quiet room, to noisy places (e.g., offices, airline terminals), and even outdoors (e.g., via the use of cellphones)
- Transducer: ranging from high-quality microphones to telephones (wireline) to cellphones (mobile) to array microphones (which track the speaker location electronically)

- Transmission channel: ranging from simple telephone channels, with μ-law speech coders in the transmission path for wireline channels, to wireless channels with fading and with a sophisticated voice coder in the path

We will see that, whenever we discuss individual recognition systems, we will characterize them in terms of this list of issues (although not every issue will be specified for each system).

26.2 Important Milestones in Speech Recognition History

Although people have been interested in machines that listen and understand for a very long time, there have been a few events that have heightened public awareness of speech recognition systems in very clear and focused ways. Perhaps the most well-publicized event was a movie that introduced the general public to machines that could understand speech and respond accordingly, namely the Stanley Kubrick movie *2001: A Space Odyssey* which was first introduced in 1968. In this movie an intelligent machine, called HAL 9000, spoke in a natural sounding voice and was able to recognize and understand fluently spoken speech, and respond intelligently. This movie raised the public awareness of the potential of machines to take over a range of tasks that interacted with humans and provided a range of services and information, on demand. A second movie that raised public awareness of the potential of machines interacting with humans was the *Star Wars* saga of George Lucas. Here the machines (in the form of the mobile robots R2D2 and C3P0) were intelligent, able to speak naturally, able to recognize and understand fluent human speech, and able to move around and interact with their environment and with other machines.

A third important milestone that occurred totally within the speech recognition community was the creation of a vision for the future, by Apple Computer, in the form of a video of a scenario of life in the year 2011 entitled *Knowledge Navigator*. This video portrayed a man working from his home and interacting with his laptop computer via an intelligent agent interface. The intelligent agent was portrayed via an animated, natural-looking face that appeared on the screen and was able to speak naturally and understand all spoken inputs. What made the video so important was the user interfaces to information that were basically defined by this video, including a speech user interface (SUI) that was as natural and easy to use as speaking to another human being, as well as a multimodal user interface (MUI) where the user had a choice of speaking, pointing and clicking, and dragging and dropping objects. The video gave clear indications of how valuable it was to integrate all these interactive modes to maximize the efficiency of interacting with the machine to achieve desired goals. This video energized the speech research community and gave it a well-defined challenge that focused technology efforts throughout the world for more than a decade.

A fourth important milestone was the introduction of a broad-based speech recognition service within the telecommunications community, called Voice Recognition Call Processing (VRCP) in 1993 by AT&T [26.1,2]. This simple voice-enabled system enabled the 100 million customers of AT&T to make a range of operator-assisted toll calls (person-to-person, third-party billing, collect, reversed charges, or operator assisted) without having to interact with a human operator. Using a simple vocabulary of five phrases (one for each of the services), users could speak any of the phrases either as an isolated command or embedded in a carrier sentence (using so-called word or phrase spotting technology), and be reliably and accurately recognized more than 99% of the time. This system ultimately properly handled about 1.2 billion calls each year and made the general public aware that more-powerful speech recognition systems were just around the corner.

A fifth important milestone was the development and introduction to the public of several speech understanding systems (both demonstrations and real-world systems) that could hold a dialogue with a human user to complete a transaction or provide some desired information. Such systems, when properly designed and implemented, appeared to be intelligent and able to handle complex tasks with seemingly minimal effort. Among such systems were the Jupiter system for weather information from MIT [26.3], the Pegasus system for flight information (also from MIT [26.4]), and the *"How May I Help You"* system for call routing of customer help line requests from AT&T [26.5,6].

Today, in 2007, there are thousands of active speech recognition systems in use in telecom environments, in automobiles, on cellphones, in home environments and

office environments, showing just how far we have come over the past 70 years. Most of these systems primarily work in limited task domains (account entry, airlines information, stock-price quotations, directions, etc.) and in limited environments (e.g., quiet office or home), but their ubiquity and performance is testimony to the degree of success that has already been achieved with the technology.

26.3 Generation 1 – The Early History of Speech Recognition

Speech research in the period between 1930 and 1950 was dominated by research which attempted to determine the relationship between the linguistic identity of a sound (the phoneme or syllable being pronounced) and the spectral characterization of that sound, the so-called acoustic–phonetic representation of speech. There was also a great deal of research in trying to understand the factors that influenced the intelligibility of speech for different sounds and various noise environments. The two most prominent researchers during this period were Harvey *Fletcher* and Homer *Dudley*, both from AT&T Bell Laboratories [26.7–9]. The pioneering work of these two speech pioneers firmly established the relationships between sound classes and signal spectrum, and showed that reliable identification of the phonetic nature of a speech sound was inherently tied in with reliable estimation of the spectral properties of that sound. The work of Fletcher, Dudley, and numerous other speech processing pioneers was quantified in the landmark book *Visible Speech* [26.10] and this book served as the bible of speech processing for a number of years.

The first generation of speech recognition research basically defined the front end analysis for modern speech recognition systems as a time-varying spectral analysis of the speech signal. Ultimately a number of variants of short-time spectral analysis were devised in subsequent years including filter bank analyses, model-based estimation methods [including the technique of linear predictive coding (LPC)], and cepstral analysis.

26.4 Generation 2 – The First Working Systems for Speech Recognition

In the second generation of research on speech recognition systems we see a number of studies trying to create realizations of the analysis of Fletcher and Dudley, namely trying to establish algorithmic approaches to the quantification of the phonetic identity of a sound (or a sequence of sounds) based on measured spectral properties of the sound, as a function of time.

Among the systems built during this period were the following:

- In 1952 *Davis*, *Biddulph*, and *Balashak* of Bell Laboratories [26.11] designed and implemented a single speaker (i. e., a speaker-trained system), isolated digit recognizer based on learned formant trajectories during the vowel regions of the spoken digit. (The formant regions are the resonances of the vocal tract in producing the sounds and generally, but not always, correspond to prominent peaks in the speech spectral density of the sound.) Even with these crude reference patterns that were based on the spectral density of the speech signal, for some speakers, virtually perfect digit recognition scores could be obtained, whereas for other speakers the digit recognition accuracy fell to the 50% range, showing the difficulty of even a simple isolated digit recognition task.
- In 1956 *Olson* and *Belar* of RCA Laboratories [26.12] built a single speaker recognition system that attempted to recognize 10 monosyllables based on a crude spectral analysis using a filter bank with eight frequency bands over the frequency range of interest. Measurements of the spectral energy of the eight bandpass filters were made at five uniformly distributed times across the spoken input (a crude form of linear time normalization) and the resulting reference pattern of eight spectral measurements and five time points was quantized to 40 bits (i. e., 1 bit for each grid point), and compared to a stored pattern for each of the 10 monosyllables. Again the performance was highly variable with talkers, but at least one talker was able to achieve 98% accuracy using this system.
- One of the first attempts to build a speaker-independent speech recognition system was de-

scribed by *Forgie* and *Forgie* of MIT Lincoln Laboratory [26.13]. The vocabulary of interest was a set of 10 vowels (spoken in the carrier syllable /b/-vowel-/t/). The front-end spectral analysis was a 35 channel filter bank from which the prominent spectral regions were estimated and the first three formants were selected to represent the vowel sound in the utterance. In limited testing the system achieved 93% accuracy, a major accomplishment at its time in history.

- In 1959 *Denes* and *Fry* at University College in London built a system to recognize a sequence of one of four vowels followed by one of nine consonants [26.14]. The system consisted of a spectral analyzer and a spectral pattern matching algorithm. Although the performance of the resulting system was not very good, the most interesting part of their research was showing that when a linguistic model based on the actual probabilities of the sequence of a given vowel, p_1, and a given consonant, p_2 were utilized in the scoring method, the performance increased from 24% accuracy to 44% accuracy – a very significant rise due to the use of a grammar.

This one-to-one association of a section of the speech signal to a distinct sound (phoneme, syllable, or word for the most part), often denoted as a segmentation and labeling approach to speech recognition, proved to be a lot more difficult than anticipated, due to the high degree of variability of the spectral information for a given sound across speakers, accents, and linguistic content of the speech being analyzed.

26.5 Generation 3 – The Pattern Recognition Approach to Speech Recognition

A number of significant technological advances in speech processing systems (and their application to speech recognition systems) occurred over the period from 1960 to 1980. In the area of spectral analysis, the method of linear predictive coding was developed by *Atal* and *Itakura* (working independently) [26.15, 16] and rapidly became the method of choice for front-end spectral analysis of speech. The concept of comparing (determining the spectral distance between) two spectral representations was proposed and studied by *Itakura* [26.17] and was widely used in pattern-based systems. The concept of optimally aligning speech spectral patterns for utterance-long sentences, based on dynamic programming concepts, has come to be known as the set of dynamic time warping (DTW) methods and was originally proposed by *Vintsyuk* in the Soviet Union [26.18] and later by *Sakoe* and *Chiba* in Japan [26.19]. The concept of quantizing a spectral representation (in the form of a vector of spectral components) using an optimally designed codebook led to the methods of vector quantization (VQ), as originally proposed by Linde et al. [26.20] and studied by *Gersho* and *Gray* [26.21]. One last important technological contribution during generation 3 was the introduction of statistical pattern clustering techniques as the method of choice for creating speaker-independent reference patterns from multiple speaker-training tokens, as proposed by *Rabiner* et al. [26.22].

The 1960s saw the introduction of several Japanese laboratories into the speech recognition arena, generally building special purpose hardware (as opposed to computer simulations) to test out their systems. Among the many Japanese systems of the 1960s were the following:

- In 1961, *Suzuki* and *Nakata*, at the Radio Research Lab in Japan, built a 26 channel spectrum analyzer as the front end for a vowel recognition system [26.23]. The channels were grouped together and connected to a set of *vowel decision circuits* (using the voiced/unvoiced and pitch period information as controls) which periodically sampled the the vowel decision outputs and choosing the phoneme most frequently recognized.
- In 1962, *Sakai* and *Doshita* of Kyoto University built a vowel/consonant recognition system based on a segmentation of speech into vowels and consonants, based on zero crossing (ZC) and frequency band energy, and obtained recognition accuracies on the order of 90% for vowels and 70% for consonants (with some phonemes eliminated from the test set) [26.24].
- In 1963, *Nagata* et al. at NEC Labs in Japan built a hardware filter bank with 8 spectral bands and created one of the first systems that was shown capable of recognizing 10 Japanese isolated digits in a highly reliable manner, using early forms of many

of the techniques that were being developed over the course of generation 3 [26.25].

There were three other key developments in speech recognition research in the 1960s that had long-lasting influence on the directions of research over a decade or more. The first was the research of Tom *Martin* and his colleagues at the RCA Laboratories [26.26]. The novel aspect of this work was the understanding that reliable detection of speech beginning and ending points enabled a rudimentary form of nonuniform time alignment between reference and test patterns, which enabled significantly more-reliable speech recognizer performance. Martin ultimately developed a set of highly reliable algorithms for word recognition, built hardware solutions that implemented these new methods, and founded one of the first companies (Threshold Technology) that built, marketed and sold speech recognition products. One of the first products from Threshold Technology, called the VIP-100 recognizer, was used for a variety of real-world applications including baggage sorting at airports, package sorting at FedEx distribution centers, and for television face plate manufacturing quality control.

The second key development of the 1960s was the pioneering research of Raj *Reddy*, initially at Stanford University and ultimately at Carnegie Mellon University, in the field of continuous speech recognition based on dynamic tracking of phonemes [26.27]. Reddy laid the foundation for more than three decades of research in speech recognition at Carnegie Mellon University and was directly involved in many of the key recognition systems of the last three generations.

The third key development at the end of the 1960s was the work of *Velichko* and *Zagoruyko* in the Soviet Union [26.28]. These researchers used the earlier work of *Vintsyuk* [26.18] to build a speech recognizer based on a pattern recognition framework and using dynamic programming time alignment methods. Velichko and Zagoruyko ultimately built a 200-word recognizer and were able to achieve reliable and robust performance for a range of talkers.

In a classic review of the state of the art in speech recognition, S. R. *Hyde* looked at the progress that had been made in speech recognition systems at the end of the 1960s, and among his many observations and conclusions were the following [26.29] (with updated perspectives):

- The debate as to the basic unit of speech recognition, namely either the phoneme or the syllable, was not yet resolved. This debate would take another 20 years to resolve, but ultimately the decision was to use context-dependent phones as the basic speech recognition unit.
- Speech cannot be directly segmented into phonemes because of the importance and the influence of context dependency. Hence all segmentation and labeling systems were essentially doomed without applying linguistic constraints over multiple phonemes.
- Time normalization based on linear shrinking or stretching the waveform to match patterns does not work, particularly when carried out over the span of a word. Speech is a dynamic signal and certain sounds of speech can be expanded or compressed almost at will, while other sounds are highly constrained and cannot easily be expanded or compressed by arbitrary amounts. Hence a suitable form of time normalization, via nonlinear time alignment methods, was crucial to the ultimate success of any recognition system.
- In order for speech recognition to work reliably in a range of speaking environments, robust spectral measurements are crucial. This observation is at the heart of current research that is seeking methods of making standard spectral measurements be robust to noise, distortion, and reverberation.
- In order to recognize speech using just acoustic measurements, an effective method of talker normalization is essential. This observation is a reflection of the spectral variability of speech as a function of the vocal tract dimensions. Hence any effective system has to either normalize out this variability or account for it as part of the recognition model. Most modern systems build statistical models which account for the variability, since it is difficult, if not impossible, to properly normalize for every possible talker.
- Devising and perfecting speech recognition systems that are effective for single talkers is of little value in solving the problem in a speaker-independent manner. Hence we need to tackle the problem of interspeaker variability directly and not try to build in fixes to the single talker systems.
- We need to utilize linguistic information in order to properly and accurately recognize fluent speech. This is basically the only way of handling the word perplexity problem and making the recognition task manageable, even for large word vocabularies.

Many of the observations made by Hyde helped guide speech recognition research for a decade or more.

26.5.1 The ARPA SUR Project

By the end of the 1960s, the Advanced Research Projects Agency (ARPA) decided that speech recognition research had advanced sufficiently that it was time to pose a challenge, namely to see whether any research lab could build a demo of a working system that could reliably and accurately recognize continuous speech. The ARPA Speech Understanding Research (SUR) program began in 1971 and had a 5 year limit for achieving a defined set of goals, with the general guidelines of:

- transforming phonemes (or syllables) into words with the aid of linguistic information
- using stress to identify semantic content words and phrase boundaries
- using syntax to constrain word sequences
- using semantics to further constrain word sequences

The specific set of goals for the ARPA SUR project were [26.30]:

> *Accept connected speech, from many cooperative speakers, in a quiet room, using a good microphone, with slight tuning/speaker, accepting* 1000 *words, using an artificial syntax, in a constraining task, yielding* $<$ 10% *semantic error, in a few times real time, on a* 100 *MIPS machine.*

One interesting aspect of the ARPA SUR project was that each participating research group was free to choose the task that was to be solved using the final recognition system.

A large number of groups participated in the SUR project, and among the technological contributions that came out of the project were the following:

- use of LPC as the front-end spectral processor became the standard for speech recognition systems [26.15, 16, 31],
- methods for reliably estimating vocal tract areas and vocal tract lengths from LPC representations [26.32],
- use of the LPC residual as a sound similarity measure [26.17],
- use of dynamic programming for time alignment [26.18, 19],
- the cepstral and cosh distance measures [26.33, 34],
- the concept of statistical modeling which later evolved into hidden Markov models [26.35],
- the importance of detecting utterance endpoints [26.36], and
- the importance of applying phonological rules of language to limit the recognition search [26.37, 38].

The major participants in the ARPA SUR project were Carnegie Mellon University (CMU), Bolt Beranek and Newman (BBN) and the System Development Corporation (SDC), working jointly with the Stanford Research Institute (SRI). A wide variety of approaches were studied during the 5 year cycle including standard segmentation and labeling methods (used in the SDC and BBN systems), along with a novel blackboard approach based on parallel asynchronous processes for proposing and verifying word hypotheses, and for proposing and verifying sentences consistent with the syntax and semantics of the chosen task (used in the CMU Hearsay II system).

Interestingly, the most successful speech recognition system, and the only one that essentially met the challenge specifications of the ARPA SUR project was the HARPY system of Bruce *Lowerre* of CMU [26.39]. This system used a conventional segmentation and labeling approach (much like systems of the earlier generation). The key novelty of the system was a compilation of the acoustic models with the lexical representation of words, the syntactic production rules for sentences, and a set of word boundary rules, into one large homogeneous network that could be readily represented as a finite state network (FSN) and searched (from left to right) using a *beam search* that only retained search paths that were close to (within a beam width) the locally optimum search path at each frame of the spoken utterance. The high degree of match between the goals of the ARPA SUR project and the performance of the HARPY system is shown in Table 26.1.

Table 26.1 Match between the goals of the ARPA SUR project and the performance of the HARPY system

Goal	HARPY
Continuous speech	Yes
Many speakers	5 (3 male, 2 female)
Cooperative	Yes
Quiet room	Computer terminal room
Good microphone	Close talking
Slight tuning/talker	20 training sentences/talker
$>$ 1000 words	1011 words
Artificial syntax	Average branching factor = 33
Constrained task	Document retrieval
$<$ 10% semantic error	5% semantic error
Few times real time	80 times real time
100 MIPS machine	0.4 MIPS PDP-KA10

Table 26.2 Performance of the various ARPA SUR systems at different tasks

System	Sentence accuracy (%)	Word branching factor	Task	Typical sentence
CMU HARPY	95	33	Document retrieval	How many articles on psychology are there?
CMU Hearsay-II	91, 74	33, 46	Document retrieval	How many articles on psychology are there?
BBN HWIM	44	195	Travel budgets	What is the plane fare to Ottawa?
SDC	24	105	Ship facts	How fast is the Theodore Roosevelt?

Table 26.2 shows a summary of the performance of two CMU systems (HARPY and HEARSAY II), the BBN *Hear What I Mean* (HWIM) system, and the SDC system. It can be seen that the performance of these speech understanding systems is a strong function of the vocabulary average word branching factor, going from a high of 95% semantic accuracy for HARPY with a word branching factor of 33 on the document retrieval task, to 24% semantic accuracy for the SDC system with a word branching factor of 105 on a ship database task. Interestingly the BBN HWIM system for travel budgets had a word branching factor of 195 but was able to achieve 44% semantic accuracy. The performance of the Hearsay II, HWIM, and SDC recognizers all failed to meet the ARPA SUR performance goal. However, the use of a set of parallel asynchronous processes (to simulate a set of component knowledge sources) with the use of a blackboard (to integrate the current state of knowledge about the words in the spoken sentence) was an innovative idea and one that defined the Hearsay II system as unique in its field. The BBN HWIM system also introduced new ideas into speech recognition systems including a lexical decoding network incorporating advanced phonological rules (whose goal was to improve the phoneme recognition rate by exploiting knowledge about contextual effects), a method of handling segmentation ambiguity using a lattice of alternate hypotheses, and the concept of performing word verification tests at the parametric level. Finally we note that at about the time that the ARPA project was ending, some new work was presented at CMU by Jim Baker, called the DRAGON system, based on statistical modeling of speech via hidden Markov models [26.35].

The ARPA SUR project had a huge impact on the field of speech recognition and natural language understanding. Among the long term achievements of this program were the following (taken from [26.40]):

1. established two viable architectures for speech recognition and natural language understanding including:
   - a uniform, precompiled, network representation of several levels of knowledge (acoustic, syntactic, and semantic)
   - a viable communication strategy between a set of parallel asynchronous processes via a common blackboard arrangement
2. established a network structure for representing task syntax (grammar) along with an understanding of the role of language perplexity on the performance of the recognizer for different tasks
3. established a viable control strategy for determining (via a network search) the best matching valid sentence in the language, using a left-to-right (or bottom up) search strategy mitigated by a beam search whose width was a function of the reliability of the recognition decision at each frame in time
4. established the importance of using semantics and context to reduce the size of the network search and to improve recognition system performance
5. established the value of precompiling the task syntax (at least for small to medium size tasks) and integrating the syntax into the overall network to be searched to find the best matching sentence in the language
6. established the value of a word verification strategy that enabled the system to backtrack and consider alternate paths when the word verification score was not above a threshold of reliability; this strategy also showed the clear benefits of applying phonological rules at word boundaries to account for pronunciation changes due to context
7. established the understanding that phonetic segmentation and labeling, as distinct processes in the recognition system, were not necessary for reliable word recognition in connected speech recognition tasks, e.g., connected digit recognition applications
8. established the value of statistical models of sounds (or words) via models like the hidden Markov model (HMM)
9. established LPC methods as the standard front-end processing for virtually all viable speech recognition systems

10. showed that talker normalization could be achieved with as few as 20 training sentences per new talker
11. showed that a full dialogue system, with appropriate response generation to each recognized sentence, was a viable model for human–machine communication systems.

By virtually any account, the ARPA SUR project was one of the most important events in the history of speech recognition and its achievements spurred the research agendas of most labs for more than a decade.

26.5.2 Research Outside of the ARPA Community

Outside of the ARPA community, two major efforts in speech recognition research were underway at IBM Research Lab and at AT&T Bell Labs. The two research efforts were aimed in completely different directions, with the effort at IBM striving to build a *voice typewriter* and the effort at Bell Labs striving to build speaker-independent speech recognizers that could be utilized within the telecommunications market for a range of applications such as voice dialing and command-and-control routing of phone calls.

The IBM research effort was led by Fred Jelinek and its goal was to build a system that could convert speech (initially in isolated word format, i. e., sentences spoken a word at a time with a slight pause between words, ultimately as continuous speech sentences) into a sequence of words (and punctuation marks) that could be entered into a text processing program and ultimately displayed on some output medium (computer monitor or paper) [26.41]. The resulting recognition system was essentially a speaker-trained system (requiring an enrollment period for each new talker) and was called the Tangora system [26.42]. The major technical achievements of the IBM research effort were the size of the vocabulary (virtually unlimited in size), with initial intended application being the creation of office correspondence using voice commands, and the creation of a statistically defined grammar. Further, IBM defined the structure of the recognition architecture as a statistical decoding algorithm that sought to maximize the likelihood of the recognized string over the set of all possible sentences that were valid within the language model of the task at hand. Using a statistical model to represent basic speech units, with a lexicon of word pronunciations and the statistical word grammar (called an n-gram language model because it proscribed the probability of a given word on the basis of the preceding $n-1$ words), the entire recognition problem was formulated as a maximum-likelihood decoding of the probability of a sentence (word sequence) in the language given the acoustic description of the speech signal (the spectral analysis vector representation of speech over time).

A block diagram of this fundamental approach to speech recognition is given in Fig. 26.1, which shows a sentence W being converted to a speech signal $s[n]$ via the speech production process. The speech signal is then spectrally analyzed (by the acoustic processor) giving the spectral representation, $X = (X_1, X_2, \ldots, X_L)$ for the L frames of the speech signal. The linguistic decoder solves for the maximum likelihood sentence, $\hat{W}$, that best matches X (i. e., maximizes the probability of W given X) via the Bayesian formulation:

$$\hat{W} = \arg\max_W P(W|X) \tag{26.1}$$

$$= \arg\max_W \frac{P(X|W)P(W)}{P(X)} \tag{26.2}$$

$$= \underbrace{\arg\max_W}_{\text{Step 3}} \underbrace{P_A(X|W)}_{\text{Step 1}} \underbrace{P_L(W)}_{\text{Step 2}} . \tag{26.3}$$

The maximization of (26.1) is converted to (26.2) using the Bayes rule, and since the denominator term $P(X)$ is independent of W, it can be removed leading to the three-step solution of (26.3). Here we explicitly denote the acoustic model by labeling $P(X|W)$ as $P_A(X|W)$, where A denotes the set of acoustic models of the speech units used in the recognizer, and we denote $P(W)$ as $P_L(W)$ for the language model describing the probabilities of various word combinations. The process of determining the maximum-likelihood solution is to first train (offline) the set of acoustic models so that step 1 in (26.3) can be evaluated for each speech utterance. The next step is to train (again offline) the language model for step 2, so that the probability of every word sequence that forms a valid sentence in the language model can be evaluated. Finally step 3 is the heart of the computation, namely a search through all possible combinations of words in the language model to determine the word

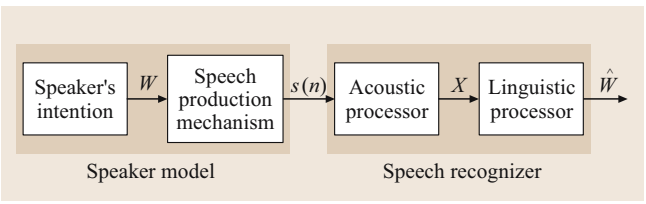


Fig. 26.1 Block diagram of the statistical model for speech recognition

sequence, W, that maximizes the probability, and finally to set the solution, $\hat{W}$ to the maximum likelihood solution obtained in step 3. Very efficient procedures for solving (26.3) have been devised [26.41, 43].

A second large effort aimed at building a speech recognition system that could readily be used by a vast population of talkers (literally tens of millions) was the focus of research at AT&T Bell Labs in the late 1970s. Inherently the system had to be *speaker independent* and had to be able to deal with a range of talkers and regional accents. This requirement led to research into methods for clustering speech patterns (from a large number of talkers) in order to create word and sound reference patterns that could be used broadly [26.22]. Since most of the telecommunications applications of interest were highly constrained tasks, with small vocabularies and only very limited language models, most of the effort was concentrated on creating highly robust acoustic models for words and subword units. Further, since the desired speech recognition systems had to work for speakers who were likely to speak carrier phrases along with the set of valid vocabulary words/phrases that were to be recognized for a given service, it was essential to develop a set of keyword spotting methods [26.44]. For example, a customer might request a credit-card service by either speaking the two-word phrase "credit card" or, alternatively, he/she might say "I'd like to make a credit card call". For highly constrained tasks, like operator services or voice dialing, it is reasonably simple to spot a small set of keywords in continuous speech and provide a highly robust service capability, rather than making the customer repeat the command without the extraneous speech.

The real significance of the speech recognition efforts at IBM and at AT&T Bell labs was that a strong mathematical formalism and rigor guided both efforts, and helped to usher in the fourth generation of speech recognition technology.

26.6 Generation 4 – The Era of the Statistical Model

The key technology development of the fourth generation of speech recognition research was a paradigm shift away from the basic pattern recognition methods that used templates and spectral distance-based time alignment of patterns, to a rigorous, mathematically based, statistical modeling framework. Two statistical methodologies evolved over the fourth generation of research, namely the HMM approach (as a complete modeling system for ASR) and the artificial neural network (ANN) approach (as a method for estimating class conditional probability densities). We discuss each of these methodologies separately.

The fundamentals of statistical modeling based on HMMs was known and understood during the third research generation by people at the Institute for Defense Analyses (IDA) in Princeton [26.45], and, by extension, the methodology and theory of the HMM was introduced at Carnegie Mellon University via the work of Jim *Baker* on the DRAGON system [26.35], and extended via the work of the IBM speech recognition research team [26.41]. To commemorate the publication and release of information about the HMM methodology [26.45], a multiday seminar was held at the IDA facilities in Princeton, NJ with attendance by people from a number of speech research laboratories. As a result of this seminar series, work was started at AT&T Bell Labs on investigating the use of HMMs for speech recognition applications in telecom, resulting in the publication of two papers by the team of *Levinson*, *Sondhi*, and *Rabiner* [26.46, 47] that explained the basic theory and gave examples of how to apply the theory to speech recognition problems. The use of HMMs in the speech recognition community grew rapidly, with strong support from the DARPA program, after the publication of these two seminal papers, and by the end of the 1980s the HMM became the preferred model of choice for speech recognition systems. Due to the steady stream of improvements and refinements of the HMM technology, use of HMMs for both automatic speech recognition and natural language understanding systems has grown in popularity and remains a vital component of all speech recognition systems today.

The strength of the HMM methodology lies in the structure of the HMM, namely the doubly stochastic process that models the intrinsic variability of the speech signal (as seen from the analyzed spectral features of each frame) as well as the structure of spoken language, in an integrated and consistent statistical modeling framework [26.48]. The HMM formalism enables a consistent modeling of the linguistic structure of speech via a Markov chain of states, and a consistent statistical description of the variability of the acoustic (spectral) features of speech via a set of probability distributions. Given a training set of a sufficient number of text-labeled

utterances, there exists within the HMM methodology an efficient estimation method, known as the Baum–Welch (or forward–backward) algorithm [26.49], for obtaining the best set of parameters that define the corresponding set of models for the speech recognition task at hand. Thus, the training phase of the HMM methodology is the estimation of the set of HMM model parameters that maximize the likelihood of the labeled utterance, given the measured spectral characterization of the speech utterance. Once the set of models is determined, via the parameter estimation procedure, the resulting models can then be used to score the likelihood of unseen speech utterances (a test set), providing a likelihood (or equivalently a probability) score that an unknown utterance is a realization of the word (or sequence of words) represented by the appropriate set of models. It rapidly became clear to the speech recognition community that the HMM approach to training and scoring of speech recognition systems represented a major leap forward from the simple pattern recognition and acoustic–phonetic methods used in earlier generations of speech recognition technology. As such, the HMM method has become the paradigm of choice for virtually all speech recognition systems since the early 1990s.

One of the strengths of the HMM methodology was its ability to incorporate a larger speech decoding framework, i. e., a language model (including both syntactic and semantic descriptions of the task grammar and semantics), represented as a finite state network (FSN). Figure 26.2 shows an example of the resulting FSN for the utterance "Show all alerts", where the individual speech sounds (context-dependent phonemes in this example) are each represented by HMMs with three state structures, silence is represented as a single state model (which can occur at the beginning and end of the utterance, as well as between words when the speaker pauses) and the grammar network for the sentence is implemented via the concatenation of the phoneme HMM and silence HMMs into a large FSN as illustrated in Fig. 26.2. An efficient method for searching any FSN in an efficient manner was created by *Mohri* in 1997 [26.50]. The resulting search method is based on a unified transducer framework for the incorporation of a weighted search and is called a finite state machine (FSM) search method.

A second key technology that was introduced into speech recognition in generation 4 was the use of neural networks. The key technological advance that brought neural network technology into the forefront (having been initially introduced in the 1940s but with little success [26.51]) was the introduction of a novel

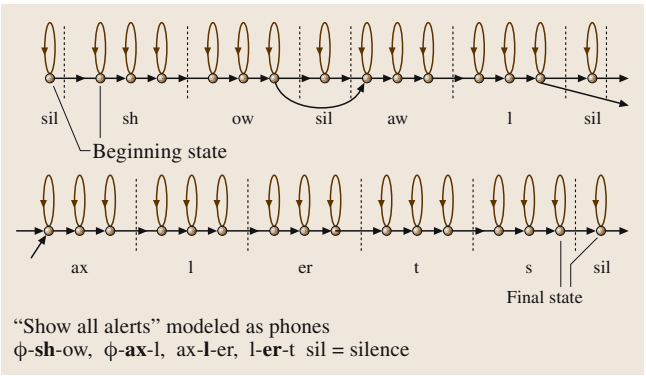


Fig. 26.2 Composite FSN for the utterance "Show all alerts"

computing structure, known as the parallel distributed processing (PDP) model which was a dense interconnection of simple neural computational elements, and a corresponding training method known as the error backpropagation method. The most popular PDP implementation for speech recognition was the multilayer perceptron (MLP), as shown in Fig. 26.3, because of its proven capability of approximating any function of the input to an arbitrary precision, provided no limitation on the complexity of the processing configuration was imposed. MLP structures were utilized in some simple speech recognition systems (e.g., recognizing isolated and connected digit strings) with good success [26.52], but tended to work poorly in trying to recognize sounds with a lot of temporal variability (e.g., stop consonants) due to the lack of any mechanism for normalizing the time sequence of the speech input accordingly. One proposed solution to this problem of handling temporal variability with neural networks was the time-delay neural network structure proposed by *Waibel* et al. [26.53], which showed that with a proper temporal structure superimposed on an MLP implementation, excellent recognition of time-varying sounds could be achieved.

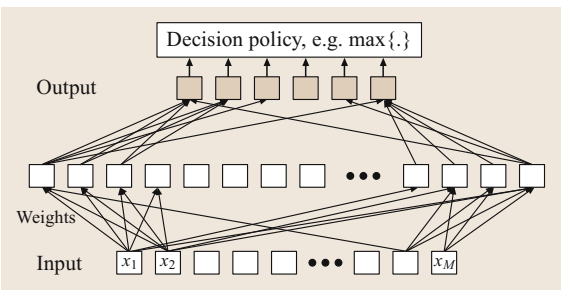


Fig. 26.3 Structure of a multilayer perceptron (MLP) with one input layer and two processing layers

Also emerging, during this period, was the method of minimum error discriminative training, which improved the performance of maximum likelihood or maximum a posteriori methods of statistical pattern recognition [26.54]. This method shifted the paradigm of estimating the data distribution for speech recognition to one that aimed to minimize the recognition error directly. While HMMs still play a crucial role in representing the individual speech units in discriminative methods, their parameters are optimized to minimize the recognition error rate rather than simply to fit the hidden Markov model (distribution) to the training data. Improvements in speech recognition accuracy have been reported in a number of tasks and applications using this set of methods [26.55, 56].

26.6.1 DARPA Programs in Generation 4

Toward the end of the 1980s, it became clear to DARPA that speech recognition technology had reached a number of key milestones and was ready to be challenged to go to the next level, namely towards a machine that could both recognize the spoken words and understand the meaning behind the recognized sentence. Further, it was felt that success with machine understanding of speech could lead to a new generation of speech technology services that could communicate with a human and perform some desired task. As such DARPA created a new, multi-year program on large vocabulary, continuous speech recognition and understanding that ultimately led to several new speech recognition and understanding systems including:

- The CMU Sphinx system [26.57], which marked the first truly speaker-independent large-vocabulary continuous speech recognition system which worked well on a range of tasks with different vocabularies, language models, language perplexities, etc. The Sphinx system successfully integrated HMMs with network search. This system was able to train and embed context-dependent phone models in a sophisticated lexical decoding network.
- The BBN BYBLOS system [26.58] which was the first HMM-based system in the DARPA program [26.59]. This system demonstrated the effectiveness of using phonetic context to improve recognition accuracy [26.60].
- The SRI DECIPHER system [26.61].

DARPA selected and defined a broad range of speech recognition and natural language understanding tasks, throughout the 1990s and into the 2000s, in order to stimulate new thinking about how to make such systems highly reliable, accurate, and robust. Regular and periodic formal evaluations of the 3 DARPA-supported systems mentioned above, along with outside systems from AT&T Bell Labs (subsequently AT&T Labs Research), Cambridge University in England, IBM, MIT, and other institutions were conducted. These formal evaluations generally measured word and sentence error rates (for both the recognition and the natural language understanding components of each system) as the performance figure of merit.

A series of tasks of increasing complexity and difficulty were devised by DARPA and were serially introduced into the speech recognition community over a period of 10–15 years. The characteristics of these tasks were the following:

Resource Management (RM). This task could be used in a military environment in order to query a ships database about the locations and properties of naval ships throughout the world. The vocabulary was about 1000 words, and the spoken queries were read from a computer-generated list of possible commands to the system. The word error rate at the end of the trials for this task was on the order of 2%, a most remarkable achievement for its time, and was achieved by almost all the systems working on this task.

Airline Travel Information System (ATIS). This task enabled the user to formulate and make travel plans via an interactive dialogue with the task (which operated using an approximation to a real airline's schedule of flights). The speech mode for this task was essentially spontaneous (allowing the user to formulate the way in which each query was made), thereby causing the recognition system to have to handle out-of-vocabulary (OOV) words without derailing the meaning of the in-vocabulary speech. The task vocabulary was about 2500 words and the word error rate, at the end of the trials for this task, was about 2.5% word error rate, again a remarkable achievement on a considerably harder task than RM.

North American Business (NAB). This task enabled the user to speak a business story from one of several sources (e.g., the Wall Street Journal). The sentences were read directly from the NAB source journal, with a resulting vocabulary of about 64 000 words. Again there was an issue with OOV words, as well as new words which appeared in the news rather frequently as events in the world changed on a day-to-day basis. However, in spite

of these difficulties, the ultimate performance on this task was a respectable 6.6% word error rate.

Broadcast News (BN). This task literally picked up over-the-air news broadcasts from the Consumer News and Business Channel (CNBC) and other news stations, and provided a running conversion from speech to text at the bottom of the TV screen. The vocabulary size was about 210 000 words and the speech was conversational (or, more likely, news read from a teleprompter). On this task the final word error rate was about 13–17%, a most respectable performance on this very difficult task.

Switchboard (SB). This task utilized speech from a standard telecom switchboard, namely conversations between two individuals over a standard telephone channel. The speech was conversational speech, of telephone quality and bandwidth, and the vocabulary size was about 45 000 words. The final word error rate on this task was in the 25–29% range, reflecting the degree of difficulty of this task.

Call Home (CH). This task utilized conversational speech from individuals calling their home telephone number, and therefore speaking to someone within their family. Such conversational speech is filled with ungrammatical snippets, short sentences, etc., due to the inherent informality in the conversation between two family members. The vocabulary for this system was about 28 000 words and the resulting word error rate was about 40%, reflecting the extreme difficulty of recognizing the colloquial level of conversational speech associated with this task.

The general conclusion that can be made from the results of this series of DARPA tasks is that conversational speech, which clearly does not strictly adhere to normal linguistic constraints, is significantly more difficult to recognize (and understand) than read speech, or speech that follows a scenario or well-defined task, i. e., speech that obeys a set of well-defined syntactic and semantic rules. Also, the DARPA program consistently showed that increasing the amount of speech data used for training each of the tasks always led to reductions in the word error rate of the system. This seems to indicate that the capacity of mixture density hidden Markov models, in representing speech for the purpose of recognition, has not been reached. Also, progress was made in the DARPA program through the use of minimum-error training methods for large-vocabulary continuous speech recognition tasks [26.55].

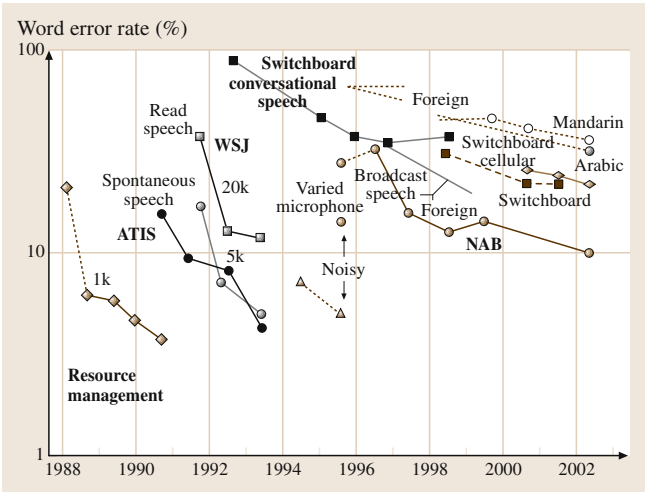


Fig. 26.4 Word error rate performance over time for a range of DARPA large-vocabulary speech recognition tasks (courtesy NIST 1999 DARPA HUB-4 Rep., *Pallett* et al. and updates from DARPA)

Figure 26.4 shows a chart that summarizes the benchmark performance of several of these large-vocabulary continuous speech recognition tasks over time [26.62]. The performance scores represent the best performance among all the submissions to standard DARPA/National Institute of Standards and Technology (NIST) evaluation tests. The most startling feature of this chart is the clear evidence that, over time, the performance of each individual task improved at almost a common rate, independent of the task complexity or task vocabulary or mode of speech input. This continuous performance improvement over time has led IBM to try to see if they can track a single task over a very long period of time and achieve performance that rivals or exceeds that of a human listener.

One fallout from the DARPA program in the 1990s and beyond was the development of sophisticated software tools that rapidly became indispensable for advanced research and development of new concepts and algorithms. Fortunately several creators of such software tools have made their source code available for use in research environments. These software toolkits for speech recognition and natural language understanding include:

1. HTK (hidden Markov model toolkit), the HMM toolkit of Cambridge University [26.63]
2. Sphinx from Carnegie Mellon University [26.64]
3. the HMM toolkit from Mississippi State University [26.65]

26.7 Generation 5 – The Future

Several important things were learned about the speech communication process between humans and machines during the first four generations of speech recognition research and the three generations of natural language understanding research. We learned that, when speaking naturally to a machine, humans tended to speak sentences that often did not obey the grammatical constraints of the recognizer (e.g., OOV words, nongrammatical constructs, ill-formed sentences, etc.). We also learned that spoken utterances were often corrupted by linguistically irrelevant *noise* (including extraneous acoustic background sounds, interfering speech, background noise, etc.). Finally we learned that, in order to successfully complete any interesting or important task, we needed to provide the tools to maintain a dialog between the user and the machine in order to reach some desired state of understanding. Such a dialog system generally required operations such as query and confirmation, thus providing some leeway for speech recognition and understanding errors. Such dialog systems have been widely used in modern speech recognition systems and some examples include the *How May I Help You* system of AT&T and the experimental demonstrations systems (Pegasus and Jupiter) developed at MIT.

We have no simple solutions to the problems that remain. Robustness to noise and other external artifacts of speaking remains a challenge that is being attacked at the signal processing level (using algorithms like spectral subtraction) [26.66], at the feature level (using algorithms like cepstral mean subtraction), and at the model level (using model adaptation methods, [26.67]), but to date none of these fixes has really solved the problem [26.68]. We look to improved models of spectral analysis, especially those related to auditory modeling [26.69], to provide some relief in this dimension, but for the foreseeable future, this remains a key challenge to ubiquitous use of speech recognition and understanding systems.

The solution to the problem of nongrammatical speech with use of OOV words remains another difficult challenge for the future. Interestingly, one possible way of handling this type of problem is the use of multiple models for recognition. For this direction, a standard generation 4 approach would first try to recognize the spoken input, providing a list of the N-best distinct sentence matches. A secondary speech model, perhaps based on a more-conventional acoustic–phonetic model of speech, might provide a set of secondary recognition scores that could be utilized whenever the word verification score of the first model falls below a reliability threshold. Systems of this type are being examined and their ability to detect and correct linguistic errors is being studied [26.70].

26.8 Summary

Figure 26.5 shows a chart that outlines the progress that has been made in speech recognition and natural language understanding technology over the past four decades. The degree of progress in technology and system complexity is clear from this chart. We see that in the 1960s we were able to recognize small vocabularies (order of 10–100 words) of isolated words, based on simple acoustic–phonetic properties of speech sounds. The key technologies that were developed during this time frame were filter bank analyses, simple time normalization methods, and the beginnings of sophisticated dynamic programming methodologies. In the 1970s we were able to recognize medium-sized vocabularies (100–1000 words) using simple template-based, pattern recognition methods. The key technologies that were developed during this period were the pattern recognition models, the introduction of LPC methods for spectral representation, the pattern clustering methods for speaker-independent systems, and the introduction of dynamic programming methods for solving connected word recognition problems. In the 1980s we started to tackle large vocabulary (1000–unlimited number of words) speech recognition problems based on statistical methods, with a wide range of networks for handling language structures. The key technologies introduced during this period were the hidden Markov model and the stochastic language model, which together enabled powerful new methods for handling virtually any continuous speech recognition problem efficiently and with high performance. In the 1990s, we were able to build large-vocabulary systems with unconstrained language models, and constrained task syntax models for continuous speech recognition and natural language understanding. The key technologies devel-

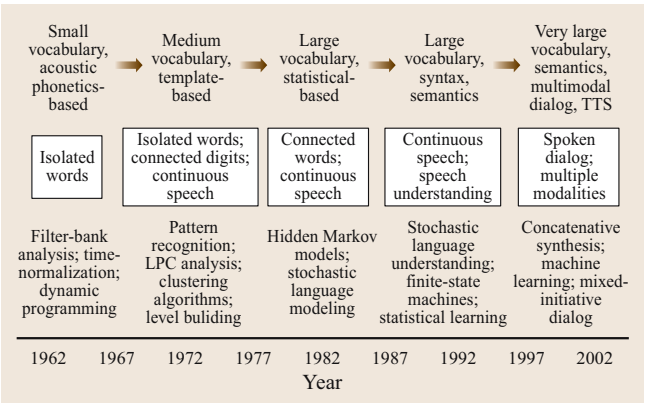


Fig. 26.5 Milestones in speech recognition and understanding technology over the past 40 years

oped during this period were the methods for stochastic language understanding, statistical learning of acoustic and language models, and the introduction of the finite state transducer framework (and the resulting FSM library) and the methods for their determination and minimization for efficient implementation of large vocabulary speech understanding systems. Finally, in the last few years, we have seen the introduction of very large vocabulary systems with full semantic models, integrated with text-to-speech (TTS) synthesis systems, and multimodal inputs (including pointing, keyboards, mice, etc.). These systems enable spoken dialog systems with a range of input and output modalities for ease-of-use and flexibility in handling adverse environments where speech might not be as suitable as other input-output modalities. During this period, we have seen the emergence of highly natural concatenative speech synthesis systems, the use of machine learning to improve both speech understanding and speech dialogs, and the introduction of mixed initiative dialog systems to enable user control of the dialog, when necessary.

The challenge of designing a machine that truly functions like an intelligent human is still a major one going forward. Our accomplishments to date are only the beginning and it will take many years before a machine can pass the Turing test, namely achieving performance that rivals that of a human.

References

26.1 J.G. Wilpon: Applications of voice-processing technology in telecommunications. In: *Voice Communications Between Humans and Machines*, ed. by D.B. Roe, J.G. Wilpon (National Academy, Washington 1994) pp. 280–310

26.2 J.G. Wilpon, D.B. Roe: AT&T telephone network applications of speech recognition, Proc. Nat. Acad. Sci. United States Am., Vol. 92 (1995)

26.3 V. Zue, S. Seneff, J. Glass, J. Polifroni, C. Pao, T. Hazen, L. Hetherington: Jupiter: A telephone-based conversational interface for weather information, IEEE Trans. Speech Audio Process. **10**, 100–112 (2000)

26.4 V. Zue, S. Seneff, J. Polifroni, M. Phillips, C. Pao, D. Goddeau, J. Glass, E. Brill: Pegasus: A spoken language interface for on-line air travel planning, Proc. Workshop on Human Language Technology, ACM Digital Library (1994) pp. 201–206

26.5 A.L. Gorin, B. Parker, R. Sachs, J.G. Wilpon: How may I help you, Proc. Interactive Voice Technology for Telecommunications Applications (IVTTA) (1996) pp. 57–60

26.6 A.L. Gorin, G. Riccardi, J.H. Wright: How may I help you, Speech Commun. **23**(1), 113–127 (1997)

26.7 H. Fletcher: The nature of speech and its interpretations, Bell Syst. Tech. J. **1**, 129–144 (1922)

26.8 H. Dudley: The vocoder, Bell Labs Rec. **17**, 122–126 (1939)

26.9 H. Dudley, R.R. Riesz, S.A. Watkins: A synthetic speaker, J. Franklin Inst. **227**, 739–764 (1939)

26.10 R.K. Potter, G.A. Kopp, H.C. Green: *Visible Speech* (van Nostrand, Amsterdam 1947)

26.11 K.H. David, R. Biddulph, S. Balashek: Automatic recognition of spoken digits, J. Acoust. Soc. Am. **24**(6), 627–642 (1952)

26.12 H.F. Olson, H. Belar: Phonetic typewriter, J. Acoust. Soc. Am. **28**(6), 1071–1081 (1956)

26.13 J.W. Forgie, C.D. Forgie: Results obtained from a vowel recognition computer program, J. Acoust. Soc. Am. **31**(11), 1480–1489 (1959)

26.14 P.B. Denes, D.B. Fry: The design and operation of the mechanical speech recognizer at university college, J. Br. Inst. Radio Eng. **19**(4), 211–229 (1959)

26.15 B.S. Atal, S.L. Hanauer: Speech analysis and synthesis by linear prediction of the speech wave, J. Acoust. Soc. Am. **50**(2), 637–655 (1971)

26.16 F. Itakura, S. Saito: A statistical method for estimation of speech spectra density and formant frequencies, Electron. Commun. Jpn. **53A**, 36–43 (1970)

26.17 F. Itakura: Minimum prediction residual principle applied to speech recognition, IEEE Trans. Acoust. Speech Signal Process. **ASSP-23**, 57–72 (1975)

26.18 T.K. Vintsyuk: Speech Discrimination by Dynamic Programming, Kibernetika **4**(2), 81–88 (1968)

26.19 H. Sakoe, S. Chiba: Dynamic programming algorithm quantization for spoken word recognition, IEEE Trans. Acoust. Speech Signal Process. **ASSP-26**(1), 43–49 (1978)

26.20 Y. Linde, A. Buzo, R.M. Gray: An algorithm for vector quantizer design, IEEE Trans. Commun. **COM-28**, 84–95 (1980)

26.21 A. Gersho, R.M. Gray: *Vector Quantization and Signal Compression* (Kluwer Academic, Boston 1991)

26.22 L.R. Rabiner, S.E. Levinson, A.E. Rosenberg, J.G. Wilpon: Speaker independent recognition of isolated words using clustering techniques, IEEE Trans. Acoust. Speech Signal Process. **ASSP-27**, 336–349 (1979)

26.23 J. Suzuki, K. Nakata: Recognition of Japanese vowels – Preliminary to the recognition of speech, J. Radio Res. Lab. **37**(8), 193–212 (1961)

26.24 T. Sakai, S. Doshita: The phonetic typewriter, Information Processing 1962, Proc. IFIP Congress (1962) pp. 445–450

26.25 K. Nagata, Y. Kato, S. Chiba: Spoken digit recognizer for Japanese language, J. Audio Eng. Soc. **12**(4), 336–342 (1964)

26.26 T.B. Martin, A.L. Nelson, H.J. Zadell: *Speech Recognition by Feature Abstraction Techniques*, Tech. Report AL-TDR-64-176 (Air Force Avionics Lab, 1964)

26.27 D.R. Reddy: *An Approach to Computer Speech Recognition by Direct Analysis of the Speech Wave*, Tech. Report No. C549 (Stanford University, 1966)

26.28 V.M. Velichko, N.G. Zagoruyko: Automatic recognition of 200 words, Int. J. Man-Machine Studies **2**, 222–234 (1970)

26.29 S.R. Hyde: Automatic speech recognition: A critical survey and discussion of the literature. In: *Human Communication: A Unified View*, ed. by E.E. David Jr., P.B. Denes (McGraw Hill, New York 1972) pp. 399–438

26.30 D.H. Klatt: Review of the ARPA speech understanding project, J. Acoust. Soc. Am. **62**, 1345–1366 (1977)

26.31 J. Makhoul: Spectral analysis of speech by linear prediction, IEEE Trans. Audio Electroacoust. **AU-21**(3), 140–148 (1973)

26.32 H. Wakita: Direct estimation of the vocal tract shape by inverse filtering of acoustic speech waveforms, IEEE Trans. Audio Electroacoust. **AU-21**(5), 417–427 (1973)

26.33 J.D. Markel, A.H. Gray Jr.: *Linear Prediction of Speech* (Springer, Berlin, Heidelberg 1976)

26.34 A.H. Gray Jr., J.D. Markel: Distance measures for speech processing, IEEE Trans. Acoust. Speech Signal Process. **ASSP-24**(5), 380–391 (1976)

26.35 J.K. Baker: The DRAGON system – An overview, IEEE Trans. Acoust. Speech Signal Process. **ASSP-23**, 24–29 (1975)

26.36 L.R. Rabiner, M.R. Sambur: An algorithm for determining the endpoints of isolated utterances, Bell Syst. Tech. J. **54**(2), 297–315 (1975)

26.37 B.T. Oshika, V. Zue, R. Weeks, H. Neu, J. Aurbach: The role of phonological rules in speech understanding research, IEEE Trans. Acoust. Speech Signal Process. **ASSP-23**(1), 104–112 (1975)

26.38 P.S. Cohen, R.L. Mercer: The phonological component of an automatic speech recognition system. In: *Speech Recognition*, ed. by D.R. Reddy (Academic, New York 1975)

26.39 B. Lowerre: The HARPY speech understanding system. In: *Readings in Speech Recognition*, ed. by A. Weibel, K.F. Lee (Morgan Kaufmann, Palo Alto 1990) pp. 576–586

26.40 G.M. White: Speech recognition: A tutorial overview, Computer **9**, 40–53 (1976)

26.41 F. Jelinek, L.R. Bahl, R.L. Mercer: Design of a linguistic statistical decoder for the recognition of continuous speech, IEEE Trans. Inform. Theory **IT-21**, 250–256 (1975)

26.42 S.K. Das, M.A. Picheny: Issues in practical large vocabulary isolated word recognition: The IBM tangora system. In: *Automatic Speech and Speaker Recognition Advanced Topics*, ed. by C.H. Lee, F.K. Soong, K.K. Paliwal (Kluwer, Boston 1996) pp. 457–479

26.43 L.R. Rabiner: A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE **77**(2), 257–286 (1989)

26.44 J.G. Wilpon, L.R. Rabiner, C.H. Lee, E.R. Goldman: Automatic recognition of keywords in unconstrained speech using hidden Markov models, IEEE Trans. Acoust. Speech Signal Process. **38**(11), 1870–1878 (1990)

26.45 J.D. Ferguson: Hidden Markov analysis: An introduction. In: *Hidden Markov Models for Speech*, ed. by J.D. Ferguson (Institute for Defense Analyses, Princeton 1980)

26.46 S.E. Levinson, L.R. Rabiner, M.M. Sondhi: An introduction to the application of the theory of probabilistic functions of a Markov process, Bell Syst. Tech. J. **62**(4), 1035–1074 (1983)

26.47 L.R. Rabiner, S.E. Levinson, M.M. Sondhi: On the application of vector quantization and hidden Markov models to speaker independent, isolated word recognition, Bell Syst. Tech. J. **62**(4), 1075–1105 (1983)

26.48 L.R. Rabiner, B.H. Juang: Statistical methods for the recognition and understanding of speech. In: *Encyclopedia of Language and Linguistics*, ed. by K. Brown, J. Lai (Elsevier, Amsterdam 2005)

26.49 L.E. Baum: An inequality and associated maximization technique in statistical estimation for

probabilistic functions of Markov processes, Inequalities **3**, 1–8 (1972)
26.50 M. Mohri: Finite-state transducers in language and speech processing, Comput. Linguist. **23**(2), 269–312 (1997)
26.51 W.S. McCullough, W.H. Pitts: A logical calculus of ideas imminent in nervous activity, Bull. Math. Biophys. **5**, 115–133 (1943)
26.52 R.P. Lippmann: Review of neural networks for speech recognition. In: *Readings in Speech Recognition*, ed. by A. Weibel, K.F. Lee (Morgan Kaufmann, Palo Alto 1990) pp. 374–392
26.53 A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, K. Lang: Phoneme recognition using time-delay neural networks, IEEE Trans. Acoust. Speech Signal Proc. **37**(3), 328–339 (1989)
26.54 B.H. Juang, S. Katagiri: Discriminative learning for minimum error classification, IEEE Trans. Signal Process. **40**(12), 3043–3054 (1992)
26.55 B.H. Juang, W. Chou, C.H. Lee: Minimum classification error rate methods for speech recognition, IEEE Trans. Speech Audio Process. **5**(3), 257–265(1997)
26.56 H. Soltau, B. Kingsbury, L. Mangu, D. Povey, G. Saon, G. Zweig: The IBM 2004 conversational telephony system for rich transcription, Proc. ICASSP, Vol. 2005 (2005) pp. I–205–I–208
26.57 K.F. Lee: *Large Vocabulary Speaker Independent Continuous Speech Recognition: The Sphinx System*, Ph.D. Thesis (Carnegie Mellon University, Pittsburgh 1988)
26.58 R. Schwartz, C. Barry, Y.-L. Chow, A. Derr, M.-W. Feng, O. Kimball, F. Kubala, J. Makhoul, J. Vandegrift: The BBN BYBLOS continuous speech recognition system, Proc. Speech and Natural Language Workshop (1989) pp. 94–99
26.59 J. Makhoul: Speech processing at BBN, IEEE Ann. Hist. Comput. **28**(1), 32–45 (2006)
26.60 R.M. Schwartz, Y.-L. Chow, O. Kimball, S. Roucos, M. Krasner, J. Makhoul: Context-dependent modeling for acoustic-phonetic recognition of continuous speech, Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (1985) pp. 1205–1208
26.61 H. Murveit, M. Cohen, P. Price, G. Baldwin, M. Weintraub, J. Bernstein: SRI's DECIPHER system, Proc. Speech and Natural Language Workshop (1989) pp. 238–242
26.62 D.S. Pallett, J. Fiscus, W. Fisher, J. Garofolo, B. Lund, A. Martin, A. Przybocki: The 1994 benchmark tests for the ARPA spoken language program, Proc. 1995 ARPA Human Language Technology Workshop (1995) pp. 5–38
26.63 S. Young: *The HTK Book* , http://htk.eng.cam.ac.uk/
26.64 Carnegie Mellon University: http://cmusphinx.soundforge.net/sphinx4/
26.65 J. Piccone: *http://www.isip.msstate.edu/projects/speech/* (Mississippi State University)
26.66 S.F. Boll: Suppression of acoustic noise in speech using spectral subtraction, IEEE Trans. Acoust. Speech Signal Process. **ASSP-27**, 113–120 (1979)
26.67 B.S. Atal: Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification, J. Acoust. Soc. Am. **55**(6), 1304–1312 (1974)
26.68 A. Acero, J. Droppo: Robustness in speech recognition. In: *Springer Handbook on Speech Processing and Speech Communication*, ed. by J. Benesty (Springer, Berlin, Heidelberg 2007)
26.69 S. Seneff: A joint synchrony/mean-rate model of auditory speech processing, J. Phonetics **16**, 55–76 (1988)
26.70 J. Hou, L.R. Rabiner, S. Dusan: Automatic speech attribute transcription (ASAT) – The front end processor. In: *Proc. Int. Conf. Acoust. Speech Signal Process.* (IEEE, 2006)

27. HMMs and Related Speech Recognition Technologies

S. Young

Almost all present-day continuous speech recognition (CSR) systems are based on hidden Markov models (HMMs). Although the fundamentals of HMM-based CSR have been understood for several decades, there has been steady progress in refining the technology both in terms of reducing the impact of the inherent assumptions, and in adapting the models for specific applications and environments. The aim of this chapter is to review the core architecture of an HMM-based CSR system and then outline the major areas of refinement incorporated into modern systems.

27.1 Basic Framework

Automatic continuous speech recognition (CSR) is sufficiently mature that a variety of real-world applications are now possible including command and control, dictation, transcription of recorded speech, and interactive spoken dialogues. This chapter describes the statistical models that underlie current-day systems: specifically, the hidden Markov model (HMM) and its related technologies.

The foundations of modern HMM-based continuous speech recognition technology were laid down in the 1970s by groups at Carnegie-Mellon, IBM, and Bell Labs [27.1–3]. Reflecting the computational power of the time, initial development in the 1980s focussed on whole-word small-vocabulary applications [27.4, 5]. In the early 1990s, attention switched to continuous speaker-independent recognition. Starting with the artificial 1000 word *resource management* task [27.6], the technology developed rapidly and by the mid-1990s, reasonable accuracy was being achieved for unrestricted dictation. Much of this development was driven by a series of Defense Advanced Research Projects Agency (DARPA) and National Security Agency (NSA) programmes [27.7], which set ever more challenging tasks, culminating most recently in systems for multilingual transcription of broadcast news programmes [27.8], and for spontaneous telephone conversations [27.9].

Although the basic framework for CSR has not changed significantly in the last 10 years, the detailed modeling techniques developed within this framework have evolved to a state of considerable sophistication (e.g., [27.10, 11]). The result has been steady and significant progress and it is the aim of this chapter to describe the main techniques by which this has been achieved. Many research groups have contributed to this progress, and each will typically have their own architectural perspective. For the sake of logical coherence, the presentation given here is somewhat biassed towards the architecture developed at Cambridge and supported by the HTK Software Toolkit [27.12] (available for free download at htk.eng.cam.ac.uk).

The chapter is organized as follows. In Sect. 27.2, the core architecture of a typical HMM-based recog-

nizer is described [27.13]. Subsequent sections then describe the various improvements that have been made to this core over recent years. Section 27.3 discusses methods of HMM parameter estimation and issues relating to improved covariance modeling. In Sects. 27.4 and 27.5, methods of normalization and adaptation are described which allow HMM-based acoustic models to more accurately represent specific speakers and environments. Finally in Sect. 27.6, the multipass architecture refinements adopted by modern transcription system is described. The chapter concludes in Sect. 27.7 with some general observations and conclusions.

27.2 Architecture of an HMM-Based Recognizer

The principal components of a large-vocabulary continuous speech recognizer are illustrated in Fig. 27.1. The input audio waveform from a microphone is converted into a sequence of fixed-size acoustic vectors $\boldsymbol{Y} = \boldsymbol{y}_1, \cdots, \boldsymbol{y}_T$ in a process called feature extraction. The decoder then attempts to find the sequence of words $W = w_1, \cdots, w_K$ that is most likely to have generated $\boldsymbol{Y}$, i. e., the decoder tries to find

$$\hat{W} = \arg\max_W [p(W|\boldsymbol{Y})] \,. \tag{27.1}$$

However, since $p(W|\boldsymbol{Y})$ is difficult to model directly, Bayes' rule is used to transform (27.1) into the equivalent problem of finding:

$$\hat{W} = \arg\max_W [p(\boldsymbol{Y}|W)p(W)] \,. \tag{27.2}$$

The likelihood $p(\boldsymbol{Y}|W)$ is determined by an *acoustic model* and the prior $p(W)$ is determined by a *language model*. In practice, the acoustic model is not normalized and the language model is often scaled by an empirically determined constant and a word-insertion penalty is added, i. e., in the log domain the total likelihood is calculated as $\log p(\boldsymbol{Y}|W) + \alpha p(W) + \beta|W|$ where α is typically in the range 8–20 and β is typically in the range 0 to −20. The basic unit of sound represented by the acoustic model is the *phone*. For example, the word *bat* is composed of three phones /b/ /ae/ /t/. About 40 such phones are required for English.

For any given W, the corresponding acoustic model is synthesized by concatenating phone models to make words as defined by a pronunciation dictionary. The parameters of these phone models are estimated from training data consisting of speech waveforms and their orthographic transcriptions. The language model is typically an N-gram model in which the probability of each word is conditioned only on its $N-1$ predecessors. The N-gram parameters are estimated by counting N-tuples in appropriate text corpora. The decoder operates by searching through all possible word sequences using pruning to remove unlikely hypotheses thereby keeping the search tractable. When the end of the utterance is reached, the most likely word sequence is output. Alternatively, modern decoders can generate lattices containing a compact representation of the most likely hypotheses.

The following sections describe these processes and components in more detail.

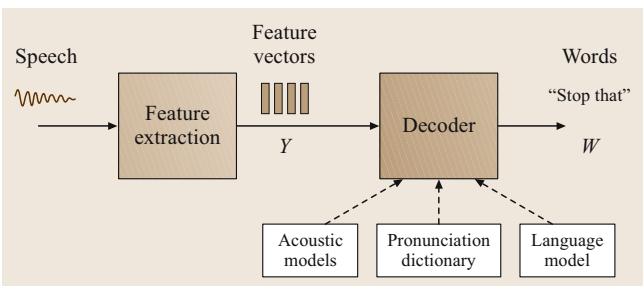


Fig. 27.1 Architecture of an HMM-based recognizer

27.2.1 Feature Extraction

The feature extraction stage seeks to provide a compact encoding of the speech waveform. This encoding should minimize the information loss and provide a good match with the distributional assumptions made by the acoustic models. Feature vectors are typically computed every 10 ms using an overlapping analysis window of around 25 ms. One of the simplest and most widely used encoding schemes uses *mel-frequency cepstral coefficients (MFCCs)* [27.14]. These are generated by applying a truncated cosine transformation to a log spectral estimate computed by smoothing an FFT with around 20 frequency bins distributed nonlinearly across the speech spectrum. The nonlinear frequency scale used is called a *mel scale* and approximates the response of the human ear. The cosine transform is applied in order to smooth the spectral estimate and decorrelate the feature elements.

Further psychoacoustic constraints are incorporated into a related encoding called *perceptual linear prediction (PLP)* [27.15]. PLP computes linear prediction coefficients from a perceptually weighted nonlinearly

compressed power spectrum and then transforms the linear prediction coefficients to cepstral coefficients. In practice, PLP can give small improvements over MFCCs, especially in noisy environments and hence it is the preferred encoding for many systems.

In addition to the spectral coefficients, first-order (delta) and second-order (delta–delta) regression coefficients are often appended in a heuristic attempt to compensate for the conditional independence assumption made by the HMM-based acoustic models. The final result is a feature vector whose dimensionality is typically around 40 and which has been partially but not fully decorrelated.

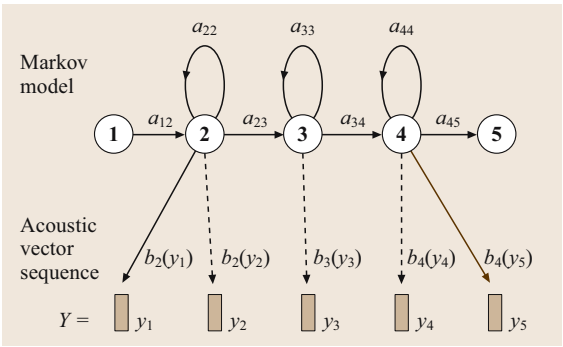


Fig. 27.2 HMM-based phone model

27.2.2 HMM Acoustic Models

As noted above, the spoken words in W are decomposed into a sequence of basic sounds called *base phones*. To allow for possible pronunciation variation, the likelihood $p(\boldsymbol{Y}|W)$ can be computed over multiple pronunciations

$$p(\boldsymbol{Y}|W) = \sum_{Q} p(\boldsymbol{Y}|Q)p(Q|W)\,. \tag{27.3}$$

Recognizers often approximate this by a max operation so that alternative pronunciations can be decoded as though they were alternative word hypotheses. Each Q is a sequence of word pronunciations $Q_1, \cdots, Q_K$ where each pronunciation is a sequence of base phones $Q_k = q_1^{(k)} q_2^{(k)} \cdots$. Then

$$p(Q|W) = \prod_{k=1}^{K} p(Q_k|w_k)\,, \tag{27.4}$$

where $p(Q_k|w_k)$ is the probability that word w_k is pronounced by base phone sequence Q_k. In practice, there will only be a very small number of possible Q_k for each w_k making the summation in (27.3) easily tractable.

Each base phone q is represented by a continuous density hidden Markov model (HMM) of the form illustrated in Fig. 27.2 with transition parameters $\{a_{ij}\}$ and output observation distributions $\{b_j()\}$. The latter are typically mixtures of Gaussians

$$b_j(\boldsymbol{y}) = \sum_{m=1}^{M} c_{jm}\mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm})\,, \tag{27.5}$$

where $\mathcal{N}$ denotes a normal distribution with mean $\boldsymbol{\mu}_{jm}$ and covariance $\boldsymbol{\Sigma}_{jm}$, and the number of components M is typically in the range 10 to 20. Since the dimensionality of the acoustic vectors $\boldsymbol{y}$ is relatively high, the covariances are usually constrained to be diagonal. The entry and exit states are *nonemitting* and they are included to simplify the process of concatenating phone models to make words.

Given the composite HMM Q formed by concatenating all of its constituent base phones, the acoustic likelihood is given by

$$p(\boldsymbol{Y}|Q) = \sum_{X} p(X, \boldsymbol{Y}|Q)\,, \tag{27.6}$$

where $X = x(0), \cdots, x(T)$ is a state sequence through the composite model and

$$p(X, \boldsymbol{Y}|Q) = a_{x(0),x(1)} \prod_{t=1}^{T} b_{x(t)}(\boldsymbol{y}_t) a_{x(t),x(t+1)}\,. \tag{27.7}$$

The acoustic model parameters $\{a_{ij}\}$ and $\{b_j()\}$ can be efficiently estimated from a corpus of training utterances using expectation maximization (EM) [27.16]. For each utterance, the sequence of base forms is found and the corresponding composite HMM constructed. A forward–backward alignment is used to compute state occupation probabilities (the E step) and the means and covariances are then estimated via simple weighted averages (the M step) [27.12, Chap. 7]. This iterative process can be initialized by assigning the global mean and covariance of the data to all Gaussian components and setting all transition probabilities to be equal. This gives a so-called *flat start* model. The number of component Gaussians in any mixture can easily be increased by cloning, perturbing the means and then re-estimating using EM.

This approach to acoustic modeling is often referred to as the *beads-on-a-string* model, so-called because all speech utterances are represented by concatenating a sequence of phone models together. The major problem with this is that decomposing each vocabulary word into

a sequence of context-independent base phones fails to capture the very large degree of context-dependent variation that exists in real speech. For example, the base form pronunciations for 'mood' and 'cool' would use the same vowel for 'oo', yet in practice the realizations of 'oo' in the two contexts are very different due to the influence of the preceding and following consonant. Context-independent phone models are referred to as *monophones*.

A simple way to mitigate this problem is to use a unique phone model for every possible pair of left and right neighbors. The resulting models are called *triphones* and, if there are N base phones, there are logically N^3 potential triphones. To avoid the resulting data sparsity problems, the complete set of *logical* triphones L can be mapped to a reduced set of physical models P by clustering and tying together the parameters in each cluster. This mapping process is illustrated in Fig. 27.3 and the parameter tying is illustrated in Fig. 27.4 where the notation x-q+y denotes the triphone corresponding to phone q spoken in the context of a preceding phone x and a following phone y. The base phone pronunciations Q are derived by simple look-up from the pronunciation dictionary; these are then mapped to logical phones according to the context, and finally the logical phones are mapped to physical models. Notice that the context dependence spreads across word boundaries and this is essential to capture many important phonological processes. For example, the [p] in *stop that* has its burst suppressed by the following consonant.

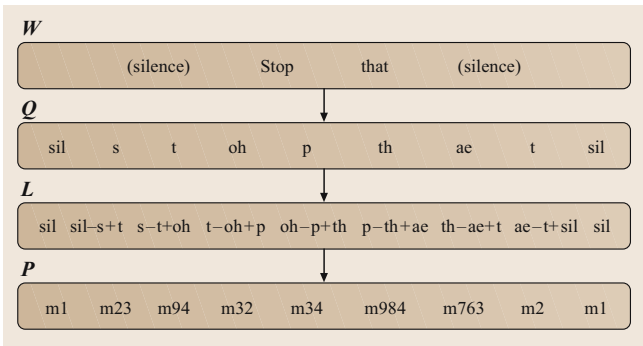


Fig. 27.3 Context-dependent phone modeling

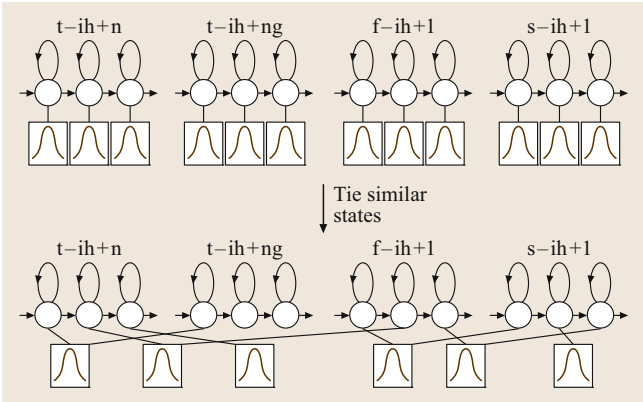


Fig. 27.4 Formation of tied-state phone models

The clustering of logical to physical models typically operates at the state level rather than the model level since it is simpler and it allows a larger set of physical models to be robustly estimated. The choice of which states to tie is made using decision trees [27.17]. Each state position of each phone q has a binary tree associated with it. Invariably each phone model has three states. Each node of the tree carries a question regarding the context. To cluster state i of phone q, all states i of all of the logical models derived from q are collected into a single pool at the root node of the tree. Depending on the answer to the question at each node, the pool of states is successively split until all states have trickled down to leaf nodes. All states in each leaf node are then tied to form a physical model. The questions at each node are selected from a predetermined set to maximize the likelihood of the training data given the final set of state tyings. If the state output distributions are single-component Gaussians and the state occupation counts are known, then the increase in likelihood achieved by splitting the Gaussians at any node can be calculated simply from the counts and model parameters without reference to the training data. Thus, the decision trees can be grown very efficiently using a greedy iterative node-splitting algorithm. Figure 27.5 illustrates this tree-based clustering. In the figure, the logical phones s-aw+n and t-aw+n will both be assigned to leaf node 3 and hence they will share the same central state of the representative physical model. The total number of tied states in a large-vocabulary speaker-independent system typical ranges between 1000 and 5000 states.

The partitioning of states using phonetically driven decision trees has several advantages. In particular, logical models that are required but were not seen at all in the training data can be easily synthesized. One disadvantage is that the partitioning can be rather coarse. This problem can be reduced using so-called *soft tying* [27.18]. In this scheme, a postprocessing stage groups each state with its one or two nearest neighbors and pools all of their Gaussians. Thus, the single Gaussian models are converted to mixture Gaussian models

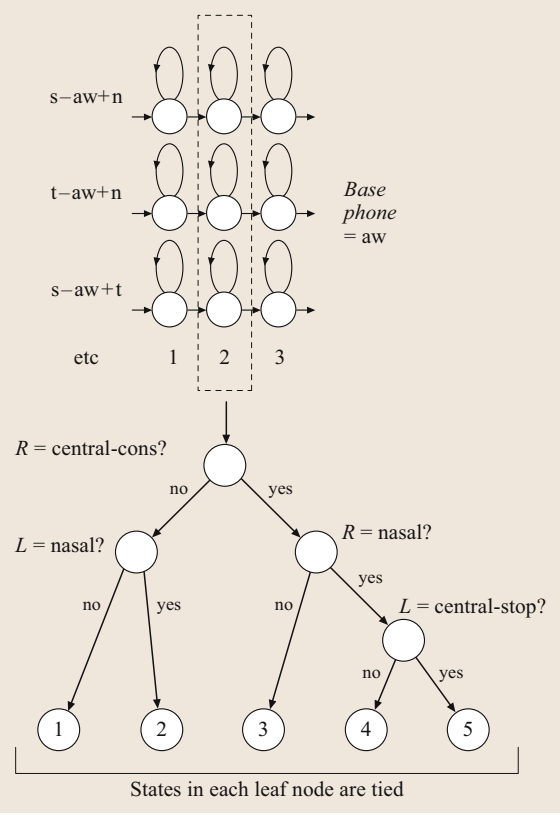


Fig. 27.5 Decision tree clustering

whilst holding the total number of Gaussians in the system constant.

To summarize, the core acoustic models of a modern speech recognizer is typically comprised of a set of tied-state mixture Gaussian HMM-based acoustic models. This core is commonly built in the following steps [27.12, Chap. 3]:

1. A flat-start monophone set is created in which each base phone is a monophone single-Gaussian HMM with means and covariances equal to the mean and covariance of the training data.
2. The parameters of the single-Gaussian monophones are iteratively re-estimated using three or four iterations of EM.
3. Each single Gaussian monophone q is cloned once for each distinct triphone x-q+y that appears in the training data.
4. The set of training data single-Gaussian triphones is iteratively re-estimated using EM and the state occupation counts of the last iteration are saved.
5. A decision tree is created for each state in each base phone, the single-Gaussian triphones are mapped into a smaller set of tied-state triphones and iteratively re-estimated using EM.
6. Mixture components are iteratively split and re-estimated until performance peaks on a held-out development set.

The final result is the required tied-state context-dependent mixture Gaussian acoustic model set.

27.2.3 *N*-Gram Language Models

The prior probability of a word sequence $W = w_1, \cdots, w_K$ required in (27.2) is given by

$$p(W) = \prod_{k=1}^{K} p(w_k | w_{k-1}, \ldots, w_1) \,. \tag{27.8}$$

For large-vocabulary recognition, the conditioning word history in (27.8) is usually truncated to $N-1$ words to form an N-gram language model

$$p(W) = \prod_{k=1}^{K} p(w_k | w_{k-1}, w_{k-2}, \ldots, w_{k-N+1}) \,, \tag{27.9}$$

where N is typically in the range 2–4. The N-gram probabilities are estimated from training texts by counting N-gram occurrences to form maximum likelihood (ML) parameter estimates. For example, let $C(w_{k-2}w_{k-1}w_k)$ represent the number of occurrences of the three words $w_{k-2}w_{k-1}w_k$ and similarly for $C(w_{k-2}w_{k-1})$, then

$$p(w_k | w_{k-1}, w_{k-2}) \approx \frac{C(w_{k-2}w_{k-1}w_k)}{C(w_{k-2}w_{k-1})} \,. \tag{27.10}$$

The major problem with this simple ML estimation scheme is data sparsity. This can be mitigated by a combination of discounting and backing-off, known as *Katz smoothing* [27.19]

$$\begin{aligned} & p(w_k | w_{k-1}, w_{k-2}) \\ & = \frac{C(w_{k-2}w_{k-1}w_k)}{C(w_{k-2}w_{k-1})} \,, && \text{if } C > C' \,, \\ & = d\frac{C(w_{k-2}w_{k-1}w_k)}{C(w_{k-2}w_{k-1})} \,, && \text{if } 0 < C \le C' \\ & = \alpha(w_{k-1}, w_{k-2}) p(w_k | w_{k-1}) && \text{otherwise} \,, \end{aligned} \tag{27.11}$$

where C' is a count threshold, d is a discount coefficient and α is a normalization constant. Thus, when the N-gram count exceeds some threshold, the ML estimate

is used. When the count is small the same ML estimate is used but discounted slightly. The discounted probability mass is then distributed to the unseen N-grams which are approximated by a weighted version of the corresponding bigram. This idea can be applied recursively to estimate any sparse N-gram in terms of a set of back-off weights and $(N-1)$-grams. The discounting coefficient is based on the Turing–Good estimate $d = (r+1)n_{r+1}/rn_r$ where n_r is the number of N-grams that occur exactly r times in the training data. Although Katz smoothing is effective, there are now known to be variations that work better. In particular, Kneser–Ney smoothing consistently outperforms Katz on most tasks [27.20, 21].

An alternative approach to robust language model estimation is to use class-based models in which, for every word w_k, there is a corresponding class c_k [27.22, 23]. Then,

$$p(W) = \prod_{k=1}^{K} p(w_k|c_k)p(c_k|c_{k-1}, \ldots, c_{k-N+1}) \,. \tag{27.12}$$

As for word-based models, the class N-gram probabilities are estimated using ML but since there are far fewer classes (typically a few hundred) data sparsity is much less of an issue. The classes themselves are chosen to optimize the likelihood of the training set assuming a bigram class model. It can be shown that, when a word is moved from one class to another, the change in perplexity depends only on the counts of a relatively small number of bigrams. Hence, an iterative algorithm can be implemented which repeatedly scans through the vocabulary, testing each word to see if moving it to some other class would increase the likelihood [27.24].

In practice it is found that for reasonably sized training sets, an effective language model for large vocabulary applications consists of a word-based trigram or four-gram interpolated with a class-based trigram.

27.2.4 Decoding and Lattice Generation

As noted in the introduction to this section, the most likely word sequence $\hat{W}$ given a sequence of feature vectors $\boldsymbol{Y} = \boldsymbol{y}_1, \cdots, \boldsymbol{y}_T$ is found by searching all possible state sequences arising from all possible word sequences for the sequence that was most likely to have generated the observed data $\boldsymbol{Y}$. An efficient way to solve this problem is to use dynamic programming. Let $\phi_j(t) = \max_X \{p(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t, x(t) = j|\mathcal{M})\}$, i. e., the maximum probability of observing the partial sequence $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_t$ and then being in state j at time t given the model $\mathcal{M}$. This probability can be efficiently computed using the Viterbi algorithm

$$\phi_j(t) = \max_i \left\{\phi_i(t-1)a_{ij}\right\} b_j(\boldsymbol{y}_t) \,. \tag{27.13}$$

It is initialized by setting $\phi_j(t)$ to 1 for the initial state and 0 for all other states. The probability of the most likely word sequence is then given by $\max_j\{\phi_j(T)\}$ and if every maximization decision is recorded, a traceback will yield the required best matching state/word sequence.

In practice, a direct implementation of the above algorithm becomes unmanageably complex for continuous speech where the topology of the models, the language model constraints, and the need to bound the computation must all be taken into account. Fortunately, much of this complexity can be abstracted away by a simple change of viewpoint.

First, the HMM topology can be made explicit by constructing a recognition network. For task-oriented applications, this network can represent the utterances that the user is allowed to say, i. e., it can represent a recognition grammar. For large-vocabulary applications, it will typically consist of all vocabulary words in parallel in a loop. In both cases, words are represented by a sequence of phone models as defined by the pronunciation dictionary (Fig. 27.6), and each phone model consists of a sequence of states as indicated by the inset. If a word has several pronunciations as in the general case described by (27.3), they are simply placed in parallel.

Given this network, at any time t in the search, a single hypothesis consists of a path through the network representing an alignment of states with feature vectors $\boldsymbol{y}_1, \cdots, \boldsymbol{y}_t$, starting in the initial state and ending

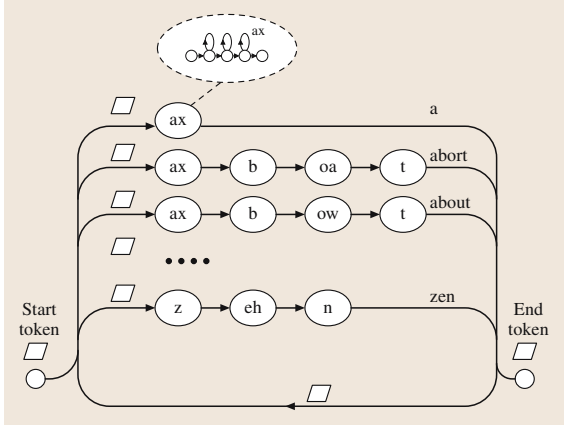


Fig. 27.6 Basic recognition network

```
Put a start token <log(1), 0> in network entry node;
Put null tokens <log(0), 0> in all other nodes;
for each time t = 1 to T do
    – word internal token propagation
    for each non-entry node j do
        maxP = log(0);
        for each predecessor node i do
            Temp token Q = Q_i;
            Q.logP += log(a_ij) [+ log(b_j(yt)) if j emitting];
            If Q.logP > maxP then
                Q_j = Q; maxP = Q.logP;
        end;
    end;
    Copy tokens from word internal exits to following entries;
    – word external token from propagation
    for each word w with entry node j do
        maxP = log(0);
        for each predecessor word u with exit node i do
            Temp token Q = Q_i;
            Q.logP += α logp(w/u) + β;
            If Q.logP > maxP then
                Q_j = Q; maxP = Q.logP; u' = u
        end;
        – Record word boundary decision
        Create a record R;
        R.Q = Q_j; Rt = t; R word = u';
        Q.link = ↑R;
    end;
    Put null token in network entry node;
end;
Token in network exit state at time T represents the best path
```

Fig. 27.7 Basic token-passing algorithm

at state j, and having a log likelihood of $\log \phi_j(t)$. This path can be made explicit via the notion of a *token* consisting of a pair of values $\langle \log P, \text{link} \rangle$, where $\log P$ is the log likelihood (often referred to as the *score*) and link is a pointer to a record of history information [27.25]. Each network node corresponding to a HMM state can store a single token and recognition proceeds by propagating these tokens around the network.

The basic Viterbi algorithm given above can now be recast for continuous speech recognition as the *token-passing* algorithm shown in outline in Fig. 27.7. The term *node* refers to a network node corresponding to a single HMM state. These nodes correspond to either entry, emitting or exit states. Essentially, tokens are passed from node to node and at each transition the token score is updated.

When a token transitions from the exit of a word to the start of the next word, its score is updated by the language model probability plus any word-insertion penalty. At the same time the transition is recorded in a record R containing a copy of the token, the current time, and the identity of the preceding word. The *link* field of the token is then updated to point to the record R. As each token proceeds through the network it accumulates a chain of these records. The best token at time T in a valid network exit node can then be examined and traced back to recover the most likely word sequence and the boundary times.

The above token-passing algorithm and associated recognition network is an exact implementation of the dynamic programming principle embodied in (27.13). To convert this to a practical decoder for speech recognition, the following steps are required:

1. For computational efficiency, only tokens that have some likelihood of being on the best path should be propagated. Thus, at every propagation cycle, the log probability of the most likely token is recorded. All tokens whose probabilities fall more than a constant below this are deleted. This results in a so-called *beam search* and the constant is called the *beam width*.
2. As a consequence of the beam search, 90% of the computation is actually spent on the first two phones of every word, after which most of the tokens fall outside of the beam and are pruned. To exploit this, the recognition network should be *tree-structured* so that word initial phones are shared (Fig. 27.8).
3. However, sharing initial phones makes it impossible to apply an exact language model probability during word-external token propagation since the identity of the following word is not known. Simply delaying the application of the language model until the end of the following word is not an option since that would make pruning ineffective. Instead, an incremental approach must be adopted in which the language model probability is taken to be the maximum possible probability given the set of possible following words. As tokens move through the tree-structured word graph, the set of possible next words reduces at each phone transition and the lan-

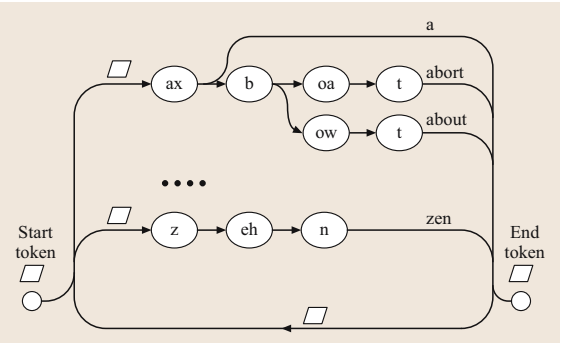


Fig. 27.8 Tree-structured recognition network

guage model probability can be updated with a more accurate estimate.

4. The HMMs linked into the recognition network should be context dependent and for best performance, this dependency should span word boundaries. At first sight this requires a massive expansion of the network, but in fact a compact static network representation of cross-word triphones is possible [27.26].
5. The dynamic programming principle relies on the principle that the optimal path at any node can be extended knowing only the state information given at that node. The use of N-gram language models causes a problem here since unique network nodes would be needed to distinguish all possible $N-1$ word histories and for large-vocabulary decoders this is not tractable. Thus, the algorithm given in Fig. 27.7 will only work for bigram language models. A simple way to solve this problem is to store multiple tokens in each state, thereby allowing paths with differing histories to *stay alive* in parallel. Token propagation now requires a merge and sort operation that, although computationally expensive, can be made tractable.

The above description of a large-vocabulary decoder covers all of the essential elements needed to recognize continuous speech in real time using just a single pass over the data. For offline batch transcription of speech, significant improvements in accuracy can be achieved by performing multiple passes over the data. To make this possible, the decoder must be capable of generating and saving multiple recognition hypotheses. A compact and efficient structure for doing this is the *word lattice* [27.27–29].

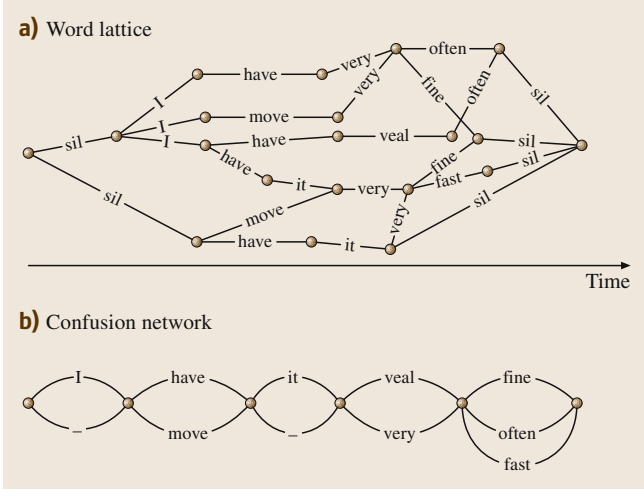


Fig. 27.9 Example lattice and confusion network

A word lattice consists of a set of nodes representing points in time and a set of spanning arcs representing word hypotheses. An example is shown in Fig. 27.9a. In addition to the word identities (IDs) shown in the figure, each arc can also carry score information such as the acoustic and language model scores. Lattices are generated via the mechanism for recording word-boundary information outlined in Fig. 27.7, except that instead of recording just the best token that is actually propagated to following word entry nodes, all word-end tokens are recorded. For the simple single-token Viterbi scheme, the quality of lattices generated in this way will be poor because many of the close-matching second-best paths will have been pruned by exercising the dynamic programming principle. The multiple-token decoder does not suffer from this problem, especially if it is used with a relatively short-span bigram language model.

Lattices are extremely flexible. For example, they can be rescored by using them as an input recognition network and can be expanded to allow rescoring by a higher-order language model. They can also be compacted into a very efficient representation called a *confusion network* [27.30, 31]. This is illustrated in Fig. 27.9b where the – arc labels indicate null transitions. In a confusion network, the nodes no longer correspond to discrete points in time; instead they simply enforce word sequence constraints. Thus, parallel arcs in the confusion network do not necessarily correspond to the same acoustic segment. However, it is assumed that most of the time the overlap is sufficient to enable parallel arcs to be regarded as competing hypotheses. A confusion network has the property that, for every path through the original lattice, there exists a corresponding path through the confusion network. Each arc in the confusion network carries the posterior probability of the corresponding word w. This is computed by finding the *link probability* of w in the lattice using a forward–backward procedure, summing over all occurrences of w and then normalizing so that all competing word arcs in the confusion network sum to one. Confusion networks can be used for minimum-word-error decoding, to provide confidence scores, and for merging the outputs of different decoders [27.32–35] (Sect. 27.6).

Finally, it should be noted that all of the above relates to one specific approach to decoding. If simple Viterbi decoding was the only requirement, then there

would be little variation amongst decoder implementations. However, the requirement to support cross-word context-dependent acoustic models and long-span language models has led to a variety of design strategies. For example, rather than have multiple tokens, the network state can be dynamically expanded to represent explicitly the currently hypothesized cross-word acoustic and long-span language model contexts [27.36, 37]. These dynamic network decoders are more flexible than static network decoders, but they are harder to implement efficiently. Recent advances in weighted finite-state transducer technology offer the possibility of integrating all of the required information (acoustic models, pronunciation, language model probabilities, etc.) into a single, very large, but highly optimized network) [27.38]. This approach offers both flexibility and efficiency and is therefore extremely useful for both research and practical applications.

A completely different approach to decoding is to avoid the breadth-first strategy altogether and use a depth-first strategy. This gives rise to a class of recognizers called *stack decoders*. Stack decoders were popular in the very early developments of ASR since they can be very efficient. However, they require dynamic networks and their run-time search characteristics can be difficult to control [27.39, 40].

27.3 HMM-Based Acoustic Modeling

The key feature of the HMM-based speech recognition architecture described in the preceding section is the use of diagonal-covariance multiple-component mixture Gaussians for modeling the spectral distributions of the speech feature vectors. If speech really did have the statistics assumed by these HMMs and if there was sufficient training data, then the models estimated using maximum likelihood would be optimal in the sense of minimum variance and zero bias [27.41]. However, since this is not the case, there is scope for improving performance both by using alternative parameter-estimation schemes and by improving the models. In this section, both of these aspects of HMM design will be discussed. Firstly, discriminative training is described and then methods of improved covariance modeling will be explored.

27.3.1 Discriminative Training

Standard maximum-likelihood training attempts to find a parameter set λ that maximizes the log likelihood of the training data, i. e., for training sentences $\boldsymbol{Y}_1, \cdots, \boldsymbol{Y}_R$, the objective function is

$$\mathcal{F}_{\mathrm{ML}}(\lambda) = \sum_{r=1}^{R} \log p_\lambda(\boldsymbol{Y}_r | \mathcal{M}_{W_r}) \,, \tag{27.14}$$

where W_r is the word sequence given by the transcription of the r-th training sentence and $\mathcal{M}_{W_r}$ is the corresponding composite HMM synthesized by concatenating phone models (denoted by Q in Sect. 27.2.2). This objective function can be maximized straightforwardly using a version of EM known as the Baum–Welch algorithm [27.42]. This involves iteratively finding the probability of state-component occupation for each frame of training data using a forward–backward algorithm, and then computing weighted averages. For example, defining the following *counts*

$$\Theta^r_{jm}(\mathcal{M}) = \sum_{t=1}^{T_r} \gamma^r_{jm}(t) \boldsymbol{y}^r_t \Big|_{\mathcal{M}} \,, \tag{27.15}$$

and

$$\Gamma^r_{jm}(\mathcal{M}) = \sum_{t=1}^{T_r} \gamma^r_{jm}(t) \Big|_{\mathcal{M}} \,, \tag{27.16}$$

where $\gamma^r_{jm}(t)$ is the probability of the model occupying mixture component m in state j at time t given the training sentence $\boldsymbol{Y}_r$ and model $\mathcal{M}$, then the updated mean estimate is given by ML as

$$\hat{\boldsymbol{\mu}}_{jm} = \frac{\sum_{r=1}^{R} \Theta^r_{jm}(\mathcal{M}_{W_r})}{\sum_{r=1}^{R} \Gamma^r_{jm}(\mathcal{M}_{W_r})} \,, \tag{27.17}$$

i. e., the average of the sum of the data weighted by the model component occupancy.

The key problem with the ML objective function is that it simply fits the model to the training data and takes no account of the model's ability to discriminate. An alternative objective function is to maximize the conditional likelihood using the maximum mutual

information (MMI) criterion [27.41, 43]

$$\mathcal{F}_{\mathrm{MMI}}(\lambda) = \sum_{r=1}^{R} \log \frac{p_\lambda(\boldsymbol{Y}_r|\mathcal{M}_{W_r})p(W_r)}{\Sigma_W p_\lambda(\boldsymbol{Y}_r|\mathcal{M}_W)p(W)} \,. \quad (27.18)$$

Here the numerator is the likelihood of the data given the correct word sequence W_r whilst the denominator is the total likelihood of the data given all possible word sequences W. Thus, the objective function is maximized by making the correct model sequence likely and all other model sequences unlikely. It is therefore discriminative. Note also that it incorporates the effect of the language model and hence more closely represents recognition conditions.

There is no simple EM-based re-estimation scheme for (27.18), however, there is an approximate scheme known as the extended Baum–Welch algorithm in which stability is achieved for the parameter updates by adding a constant D to both the numerator and denominator. For example, (27.17) in the extended scheme becomes

$$\hat{\boldsymbol{\mu}}_{jm} = \frac{\sum_{r=1}^{R}[\Theta^r_{jm}(\mathcal{M}_{\mathrm{num}}) - \Theta^r_{jm}(\mathcal{M}_{\mathrm{den}})] + D\boldsymbol{\mu}_{jm}}{\sum_{r=1}^{R}[\Gamma^r_{jm}(\mathcal{M}_{\mathrm{num}}) - \Gamma^r_{jm}(\mathcal{M}_{\mathrm{den}})] + D} \,, \quad (27.19)$$

where $\mathcal{M}_{\mathrm{num}}$ is the combined acoustic and language model used to compute the numerator of (27.18), and $\mathcal{M}_{\mathrm{den}}$ is the combined acoustic and language model used to compute the denominator. In the latter case, it is understood that the counts in $\Theta^r_{jm}(\mathcal{M}_{\mathrm{den}})$ are summed over all word sequences. For large-vocabulary continuous speech, this is approximated by computing lattices and summing over all lattice arcs. Although the numerator counts can be computed as in ML, in practice, the numerator counts are also computed using lattices since this provides a convenient way to take account of multiple pronunciations [27.44]. As can be seen, counts in the numerator are reduced if there are similar confusing counts in the denominator. The constant D acts like an interpolation weight between the new estimate and the existing estimate.

In a little more detail, the MMI training process is as follows. Numerator and denominator lattices are generated for every training utterance using $\mathcal{M}_{\mathrm{num}}$ and $\mathcal{M}_{\mathrm{den}}$, respectively. $\mathcal{M}_{\mathrm{num}}$ consists of the current phone models integrated into a graph of alternative word pronunciations, and $\mathcal{M}_{\mathrm{den}}$ consists of the normal recognizer set-up with two exceptions. Firstly, a weaker language model is used. Typically the lattices are generated using a bigram language model and then rescored using a unigram [27.45]. Secondly, the likelihoods of the acoustic models are scaled by the inverse of the normal LM scaling factor [27.46]. Both of these modifications have been found to increase the number of confusions in the denominator lattice, thereby improving subsequent generalization. Once the word-level lattices have been generated, a Viterbi decode is performed on each lattice arc to obtain a phone-level segmentation. Forward–backward is then applied to obtain the component level counts and the model parameters are re-estimated using (27.19) (and similar formulae for the variances and mixture weights). This process is iterated and the state-component alignments are recomputed every three or four iterations. The word-level lattices are typically held fixed throughout the training process.

The constant D must be chosen to be large enough to ensure convergence but small enough to ensure acceptably fast training. In practice, D is chosen to ensure that variance updates are positive and is normally set specifically for each phone or Gaussian [27.46].

MMI training can provide consistent performance improvements compared to similar systems trained with ML [27.46]. However, it can be argued that it places too much emphasis on training utterances that have a low a posterior probability and not enough weight on training utterances near the decision boundary, as in for example minimum classification error (MCE) training [27.47]. MCE, however, focuses on overall sentence-level accuracy, and it is not appropriate for lattice-based training of large systems. Minimum-phone-error (MPE) training addresses this issue by attempting to maximize the a posterior utterance probability scaled by the *raw phone accuracy* (RPA) [27.48]

$$\mathcal{F}_{\mathrm{MPE}}(\lambda) = \sum_{r=1}^{R} \frac{\Sigma_W p_\lambda(\boldsymbol{Y}_r|\mathcal{M}_W)p(W)\mathrm{RPA}(W, W_r)}{\Sigma_W p_\lambda(\boldsymbol{Y}_r|\mathcal{M}_W)p(W)} \,, \quad (27.20)$$

where, as in lattice-based MMI training, the sums over W are taken over all word sequences appearing in the lattice generated by $\mathcal{M}_{\mathrm{den}}$. The RPA is a measure of the number of phones accurately transcribed in each word string hypothesis W. Given the times of the phone boundaries, each phone in W is matched against the corresponding time segment in the transcription W_r. If the phones are the same, the RPA is incremented by the percentage overlap; otherwise it is decremented by the

percentage overlap. Parameter optimization is similar to the MMI process described above except that the counts are only computed on the denominator lattice. The numerator lattice provides the transcriptions needed for determining the RPA. Essentially, the counts are composed from the occupation probabilities scaled by the RPA. If they are positive, they contribute to the numerator terms in the update equations, and if they are negative, they contribute to the denominator terms (see [27.48] for details).

The generalization capabilities of discriminatively trained systems can be improved by interpolating with ML. For example, the H-criterion interpolates objective functions: $\alpha\mathcal{F}_{\text{MMI}} + (1-\alpha)\mathcal{F}_{\text{ML}}$ [27.49]. However, choosing an optimal value for α is difficult and the effectiveness of the technique decreases with increasing training data [27.50]. A more-effective technique is *I-smoothing* which increases the weight of the numerator counts depending on the amount of data available for each Gaussian component [27.48]. This is done by scaling the numerator counts $\Gamma^r_{jm}(\mathcal{M}_{\text{num}})$ and $\Theta^r_{jm}(\mathcal{M}_{\text{num}})$ by

$$1 + \frac{\tau}{\Sigma_r \Gamma^r_{jm}(\mathcal{M}_{\text{num}})} , \tag{27.21}$$

where τ is a constant (typically about 50). As τ increases from zero, more weight is given to ML.

27.3.2 Covariance Modeling

An alternative way of improving the acoustic models is to allow them to more closely match the true distribution of the data. The baseline acoustic models outlined in Sect. 27.2.2 use mixtures of diagonal-covariance Gaussians chosen as a compromise between complexity and modeling accuracy. Nevertheless, the data is clearly not diagonal and hence finding some way of improving the covariance modeling is desirable. In general the use of full-covariance Gaussians in large-vocabulary systems would be impractical due to the sheer size of the model set although given enough data it can be done [27.11]. However, the use of shared or *structured* covariance representations allow covariance modeling to be improved with very little overhead in terms of memory and computational load.

One the simplest and most effective structured covariance representations is the semi-tied covariance (STC) matrix [27.51]. STC models the covariance of the m-th Gaussian component as

$$\hat{\Sigma}_m = A^{-1}\Sigma_m(A^{-1})^T , \tag{27.22}$$

where Σ_m is the component-specific diagonal covariance and A is the STC matrix shared by all components. If the component mean is $\boldsymbol{\mu}_m = A\hat{\boldsymbol{\mu}}_m$ then component likelihoods can be computed by

$$\mathcal{N}(\boldsymbol{y}; \hat{\boldsymbol{\mu}}_m, \hat{\Sigma}_m) = |A|\mathcal{N}(A\boldsymbol{y}; \boldsymbol{\mu}_m, \Sigma_m) . \tag{27.23}$$

Hence, a single STC matrix can be viewed as a linear transform applied in feature space. The component parameters $\boldsymbol{\mu}_m$ and Σ_m represent the means and variances in the transformed space and can be estimated in the normal way by simply transforming the training data. The STC matrix itself can be efficiently estimated using a row-by-row iterative scheme. Furthermore it is not necessary during training to store the full-covariance statistics at the component level. Instead, an interleaved scheme can be used in which the A matrix statistics are updated on one pass through the training data, and then the component parameters are estimated on the next pass [27.51]. The means can in fact be updated on both passes. This can be integrated into the *mixing-up* operation described in Sect. 27.2.2. For example, a typical training scheme might start with a single Gaussian system and an identity A matrix. The system is then iteratively refined by re-estimating the component parameters, updating the A matrix, and mixing up until the required number of Gaussians per state is achieved. As well as having a single global A matrix, the Gaussian components can be clustered and assigned one A matrix per cluster. For example, there could be one A matrix per phone or per state depending on the amount of training data available and the acceptable number of parameters. Overall, the use of STC can be expected to reduce word error rates by around 5–10% compared to the baseline system. In addition to STC, other types of structured covariance modeling include factor-analyzed HMMs [27.52], subspace-constrained precision and means (SPAM) [27.53], and extended maximum likelihood linear transform (EMLLT) [27.54].

It can be shown [27.51] that simultaneous optimization of the full set of STC parameters (i. e., $\{A, \boldsymbol{\mu}_m, \Sigma_m\}$) is equivalent to maximizing the auxiliary equation

$$\mathcal{Q}_{\text{STC}} = \sum_{t,m} \gamma_m(t) \log\left(\frac{|A|^2}{|\text{diag}(A\, W^{(m)} A^{\text{T}})|}\right) , \tag{27.24}$$

where

$$W^{(m)} = \frac{\Sigma_t \gamma_m(t)\bar{\boldsymbol{y}}_m(t)\bar{\boldsymbol{y}}_m(t)^{\text{T}}}{\Sigma_t \gamma_m(t)} \tag{27.25}$$

and where $\bar{\boldsymbol{y}}_m(t) = \boldsymbol{y}(t) - \hat{\boldsymbol{\mu}}_m$. If each Gaussian component is regarded as a class, then $W^{(m)}$ is the within-class covariance and it can be shown that (27.24) is the maximum-likelihood solution to a generalized form of linear discriminant analysis called heteroscedastic LDA (linear discriminant analysis) (HLDA) in which class covariances are not constrained to be equal [27.55]. The matrix A can therefore be regarded as a feature space transform that discriminates Gaussian components and this suggests a simple extension by which A can also perform a subspace projection, i. e.,

$$\hat{\boldsymbol{y}} = A\boldsymbol{y} = \begin{pmatrix} A_{[p]}\boldsymbol{y} \\ A_{[d-p]}\boldsymbol{y} \end{pmatrix} = \begin{pmatrix} \hat{\boldsymbol{y}}_{[p]} \\ \hat{\boldsymbol{y}}_{[d-p]} \end{pmatrix} , \qquad (27.26)$$

where the d-dimensional feature space is divided into p *useful* dimensions and $d-p$ *nuisance* dimensions. The matrix $A_{[p]}$ projects $\boldsymbol{y}$ into a p-dimensional subspace that is modeled by the diagonal Gaussian mixture components of the acoustic models. The $d-p$ nuisance dimensions are modeled by a global nondiscriminating Gaussian. Equation (27.24) can therefore be factored as

$$\mathcal{Q}_{\text{HLDA}} = \sum_{t,m} \gamma_m(t) \times \log\left(\frac{|A|^2}{|\text{diag}(A_{[p]} W^{(m)} A_{[p]}^T)||\text{diag}(A_{[d-p]} T A_{[d-p]}^T)|} \right) , \qquad (27.27)$$

where T is the global covariance of the training data. The forms of (27.24) and (27.27) are similar, and the optimal value for $A_{[p]}$ can be estimated by the same row-by-row iteration used in the STC case.

For application to large vocabulary continuous speech recognition (LVCSR), $\boldsymbol{y}$ can be constructed either by concatenating successive feature vectors, or as is common in HTK-based systems, the standard 39-element feature vector comprised of static PLP coefficients plus their first and second derivatives is augmented by the third derivatives and then projected back into 39 dimensions using a 39×52 HLDA transform.

Finally note that, as with semi-tied covariances, multiple HLDA transforms can be used to allow the full acoustic space to be covered by a set of piecewise linear projections [27.56].

27.4 Normalization

Normalization attempts to condition the incoming speech signal to minimize the effects of variation caused by the environment and the physical characteristics of the speaker.

27.4.1 Mean and Variance Normalization

Mean normalization removes the average feature value and since most front-end feature sets are derived from log spectra, this has the effect of reducing sensitivity to channel variation. Cepstral variance normalization scales each individual feature coefficient to have a unit variance and empirically this has been found to reduce sensitivity to additive noise [27.57].

For transcription applications where multiple passes over the data are possible, the necessary mean and variance statistics should be computed over the longest possible segment of speech for which the speaker and environment conditions are constant. For example, in broadcast news transcription this will be a speaker segment and in telephone transcription it will be a whole side of a conversation. Note that for real-time systems that operate in a single, continuous pass over the data, the mean and variance statistics must be computed as running averages.

27.4.2 Gaussianization

Given that normalizing the first- and second-order statistics yields improved performance, an obvious extension is to normalize the higher-order statistics so that the features are Gaussian distributed. This so-called *Gaussianization* is performed by finding a transform $z = \phi(y)$, on a per-element basis, such that $p(z)$ is Gaussian. One simple way to achieve this is to estimate a cumulative distribution function (CDF) for each feature element y

$$F_0(y) = \frac{1}{N} \sum_{i=1}^{N} h(y - y_i) = \frac{\text{rank}(y_i)}{N} , \qquad (27.28)$$

where $y_1, \cdots, y_N$ are the data to be normalized, $h(\cdot)$ is the step function and $\text{rank}(y_i)$ is the rank of y_i when the data are sorted. The required transformation is then

$$z_i = \Phi^{-1}\left(\frac{\text{rank}(y_i)}{N} \right) , \qquad (27.29)$$

where $\Phi(\cdot)$ is the CDF of a Gaussian [27.58].

One difficulty with this approach is that when the normalization data set is small the CDF estimate can be noisy. An alternative approach is to estimate an M-component Gaussian mixture model (GMM) on the data and then use this to approximate the CDF [27.59], that is

$$z_i = \Phi^{-1}\left(\int_{-\infty}^{y_i}\sum_{m=1}^{M} c_m \mathcal{N}(y;\mu_m,\sigma_m^2)\mathrm{d}y\right) , \quad (27.30)$$

where μ_m and σ_m^2 are the mean and variance of the m-th GMM component. This results in a smoother and more-compact representation of the Gaussianization transformation [27.60].

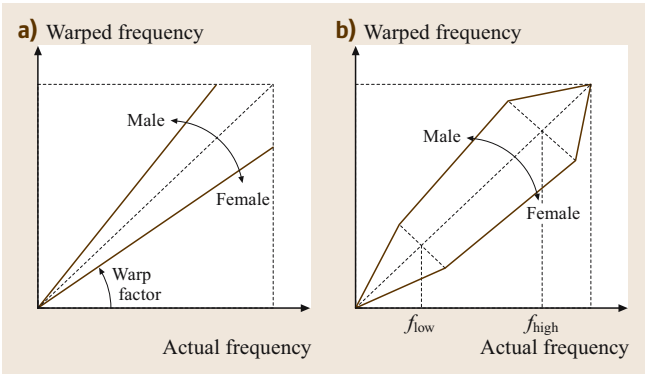


Fig. 27.10a,b Vocal-tract length normalization

27.4.3 Vocal-Tract-Length Normalization

Variations in vocal-tract length cause formant frequencies to shift in frequency in an approximately linear fashion. Thus, one simple form of normalization is to linearly scale the filter bank center frequencies within the front-end feature extractor to approximate a canonical formant frequency scaling [27.61]. This is called *vocal-tract-length normalization (VTLN)*.

To implement VTLN two issues need to be addressed: the definition of the scaling function and the estimation of the appropriate scaling function parameters for each speaker. Early attempts at VTLN used a simple linear mapping but, as shown in Fig. 27.10a, this results in a problem at high frequencies where female voices have no information in the upper frequency band and male voices have the upper frequency band truncated. This can be mitigated by using a piecewise linear function of the form shown in Fig. 27.10b [27.57]. Alternatively, a bilinear transform can be used [27.62]. Parameter estimation is performed using a grid search plotting log likelihoods against parameter values. Once the optimal values for all training speakers have been computed, the training data is normalized and the acoustic models re-estimated. This is repeated until the VTLN parameters have stabilized. Note here that, when comparing log likelihoods resulting from differing VTLN transformations, the Jacobean of the transform should strictly be included. This is however very complex to estimate and since the application of mean and variance normalization will reduce the affect of this approximation, it is usually ignored.

For very large systems, the overhead incurred from iteratively computing the optimal VTLN parameters can be considerable. An alternative is to approximate the effect of VTLN by a linear transform. The advantage of this approach is that the optimal transformation parameters can be determined from the auxiliary function in a single pass over the data [27.63].

VTLN is particularly effective for telephone speech where speakers can be clearly identified. It is less effective for other applications such as broadcast news transcription where speaker changes must be inferred from the data.

27.5 Adaptation

A fundamental idea in statistical pattern classification is that the training data should adequately represent the test data, otherwise a mismatch will occur and recognition accuracy will be degraded. In the case of speech recognition, there will always be new speakers who are poorly represented by the training data, and new hitherto unseen environments. The solution to these problems is *adaptation*. Adaptation allows a small amount of data from a target speaker to be used to transform an acoustic model set to make it more closely match that speaker. It can be used both in training to make more-specific and/or more-compact recognition sets, and it can be used in recognition to reduce mismatch and the consequent recognition errors.

There are various styles of adaptation that affect both the possible applications and the method of implementation. Firstly, adaptation can be *supervised*, in which case accurate transcriptions are available for all of the adaptation data, or it can be *unsupervised*, in which case the required transcriptions must be hypothesized. Secondly,

adaptation can be *incremental* or *batch mode*. In the former case, adaptation data becomes available in stages, for example, as is the case for a spoken dialogue system when a new caller comes on the line. In batch mode, all of the adaptation data is available from the start, as is the case in offline transcription.

This section describes the main approaches to adaptation and its application in both recognition and training.

27.5.1 Maximum A Posteriori (MAP) Adaptation

Given some adaptation data $\boldsymbol{Y}_1, \cdots, \boldsymbol{Y}_R$ and a model $\mathcal{M}$ with parameters λ, MAP-based parameter estimation seeks to maximize the following objective function,

$$\mathcal{F}_{\text{MAP}}(\lambda) = \sum_{r=1}^{R} \log p(\boldsymbol{Y}_r|\mathcal{M}_{W_r}) p(\lambda) \,. \qquad (27.31)$$

Comparing this with the ML objection function given in (27.14), it can be seen that the likelihood is weighted by the prior. The choice of distribution for this prior is problematic since there is no conjugate prior density for a continuous Gaussian mixture HMM. However, if the mixture weights and Gaussian component parameters are assumed to be independent, then a finite mixture density of the form $p_{\text{D}}(\boldsymbol{c}_j) \prod_m p_{\text{W}}(\boldsymbol{\mu}_{jm}, \Sigma_{jm})$ can be used, where $p_{\text{D}}(\cdot)$ is a Dirichlet distribution over the vector of mixture weights $\boldsymbol{c}_j$ and $p_{\text{W}}(\cdot)$ is a normal–Wishart density. It can then be shown that this leads to parameter estimation formulae of the form

$$\hat{\boldsymbol{\mu}}_{jm} = \frac{\tau \boldsymbol{\mu}_{jm} + \sum_{r=1}^{R} \Theta^r_{jm}(\mathcal{M}_{W_r})}{\tau + \sum_{r=1}^{R} \Gamma^r_{jm}(\mathcal{M}_{W_r})} \,, \qquad (27.32)$$

where $\boldsymbol{\mu}_{jm}$ is the prior mean, and τ is a parameter of $p_{\text{W}}(\cdot)$, which is normally determined empirically. Similar, though rather more-complex, formulae can be derived for the variances and mixture weights [27.64].

Comparing (27.32) with (27.17), it can be seen that MAP adaptation effectively interpolates the original prior parameter values with those that would be obtained from the adaptation data alone. As the amount of adaptation data increases, the parameters tend asymptotically to the adaptation domain. This is a desirable property and it makes MAP especially useful for porting a well-trained model set to a new domain for which there is only a limited amount of data.

A major drawback of MAP adaptation is that every Gaussian component is updated individually. If the adaptation data is sparse, then many of the model parameters will not be updated. Various attempts have been made to overcome this [27.65,66] but MAP nevertheless remains ill-suited for rapid incremental adaptation.

27.5.2 ML-Based Linear Transforms

An alternative approach to adaptation is to build a set of linear transforms to map an existing model set into a new adapted model set such that the likelihood of the adaptation data is maximized. This is called *maximum-likelihood linear regression* (MLLR) and, unlike MAP, it is particularly well suited to unsupervised incremental adaptation.

There are two main variants of MLLR: unconstrained and constrained [27.67]. In unconstrained MLLR, separate transforms are trained for the means and variances,

$$\hat{\boldsymbol{\mu}}_{jm} = G\boldsymbol{\mu}_{jm} + \boldsymbol{b}; \; \hat{\Sigma}_{jm} = H\Sigma_{jm}H^{\text{T}} \,. \qquad (27.33)$$

The formulae for constrained MLLR (CMLLR) are identical except that $G = H$. The maximum-likelihood estimation formulae are given in [27.68]. Whereas there are closed-form solutions for unconstrained MLLR, the constrained case is similar to the semi-tied covariance transform discussed in Sect. 27.3.2 and requires an iterative solution. However, CMLLR has other advantages as discussed in Sect. 27.5.3 below. Linear transforms can also be estimated using discriminative criteria [27.69–71].

The key to the power of the MLLR adaptation approach is that a single transform matrix G can be shared across a set of Gaussian mixture components. When the amount of adaptation data is limited, a global transform can be shared across all Gaussians in the system. As the amount of data increases, the HMM state components

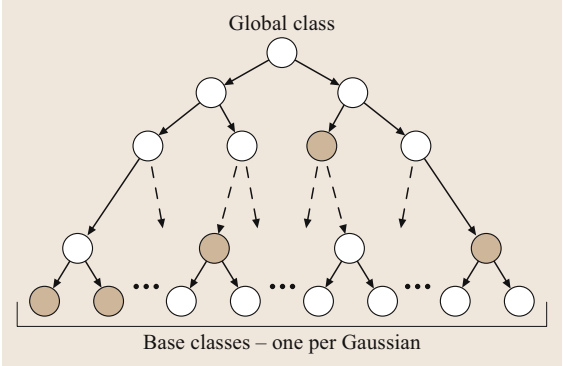


Fig. 27.11 A regression class tree

can be grouped into classes, with each class having its own transform. As the amount of data increases further, the number of classes and therefore transforms increases correspondingly, leading to increasingly good adaptation.

The number of transforms to use for any specific adaptation set can be determined automatically using a *regression class tree* as illustrated in Fig. 27.11. Each node represents a regression class, i. e., a set of Gaussian components that will share a single transform. The total occupation count associated with any node in the tree can easily be computed since the counts are known at the leaf nodes. Then, for a given set of adaptation data, the tree is descended and the most specific set of nodes for which there is sufficient data is selected (for example, the shaded nodes in the figure).

When used in the unsupervised mode, MLLR is normally applied iteratively [27.72]. First, the unknown test speech is recognized, then the hypothesized transcription is used to estimate the MLLR transforms. The test speech is then re-recognized using the adapted models. This is repeated until convergence is achieved. A refinement of this is to use recognition lattices in place of the 1-best hypothesis to accumulate the adaptation statistics. This approach is more robust to recognition errors and avoids the need to re-recognize the data since the lattice can simply be rescored [27.73].

27.5.3 Adaptive Training

Ideally, an acoustic model set should encode just those dimensions that allow the different classes to be discriminated. However, in the case of speaker-independent (SI) speech recognition, the training data necessarily includes a large number of speakers and hence acoustic models trained directly on this set will have to *waste* a large number of parameters encoding the variability between speakers rather than the variability between spoken words, which is the true aim.

One way to overcome this is to replace the single SI model set with a cluster of more-specific models where each model can be trained on more-homogenous data. This is called *cluster adaptive training* (CAT). At recognition time, a linear combination of models is selected where the set of interpolation weights, in effect, forms a speaker-specific transform [27.74–76]. More recently discriminative techniques have been applied to CAT with some success [27.77].

An alternative approach to CAT is to use adaptation to transform each training set speaker to form a canonical model. This is called *speaker adaptive training*

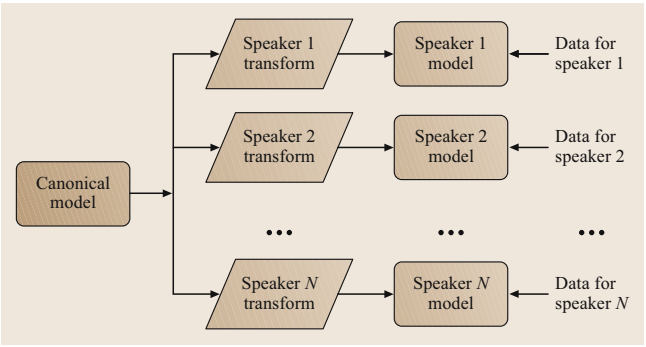


Fig. 27.12 Adaptive training

(SAT) and the conceptual schema for this is shown in Fig. 27.12 [27.78]. When only mean transformations are used, SAT is straightforward. A transform is estimated for each speaker, and then the parameters of the canonical model set are estimated by modifying the statistics to account for the transform. For example, to estimate the canonical model means, the counts in (27.15) and (27.16) are modified as

$$\Theta^r_{jm}(\mathcal{M}) = \sum_{t=1}^{T_r} \gamma^r_{jm}(t) G^{(r)T} \Sigma^{-1}_{jm} (\boldsymbol{y}^r_t - \boldsymbol{b}^{(r)}) \Big|_{\mathcal{M}} , \tag{27.34}$$

and

$$\Gamma^r_{jm}(\mathcal{M}) = \sum_{t=1}^{T_r} \gamma^r_{jm}(t) G^{(r)T} \Sigma^{-1}_{jm} G^{(r)} \Big|_{\mathcal{M}} , \tag{27.35}$$

where $G^{(r)}, \boldsymbol{b}^{(r)}$ is the transform for the speaker uttering training sentence r. The mean is then estimated using (27.17) as normal.

Rather than modifying the statistics, the use of CMLLR allows adaptive training to be simplified further and allows combined mean and variance adaptation. Similar to the case for semi-tied covariances, the CMLLR transformed likelihood can be computed simply by regarding it as a feature space transformation, i. e., for any mixture component m of state j

$$\mathcal{N}(\boldsymbol{y}; \hat{\boldsymbol{\mu}}_{jm}, \hat{\Sigma}_{jm}) = \frac{1}{|G|} \mathcal{N}(G^{-1}(\boldsymbol{y} - \boldsymbol{b}); \boldsymbol{\mu}_{jm}, \Sigma_{jm}) , \tag{27.36}$$

where $\hat{\boldsymbol{\mu}}_{jm}$ and $\hat{\Sigma}_{jm}$ are the transformed means and variances as in (27.33) (with $G = H$). Thus a SAT system can be built using CMMLR simply by iterating between the estimation of the canonical model using estimation

of the transformed training data and the transforms using the canonical model.

Finally, note that SAT trained systems incur the problem that they can only be used once transforms have been estimated for the test data. Thus, an SI model set is typically retained to generate the initial hypothesized transcription or lattice needed to compute the first set of transforms.

27.6 Multipass Recognition Architectures

The previous sections have reviewed some of the basic techniques available for both training and adapting an HMM-based recognition system. In general, any particular combination of model set and adaptation technique will have slightly different characteristics and make different errors. Furthermore, if the outputs of these systems are converted to confusion networks, as explained in Sect. 27.2.4, then it is straightforward to combine the confusion networks and then select the word sequence with the overall maximum a posterior likelihood. Thus, modern transcription systems typically utilize multiple model sets and make multiple passes over the data.

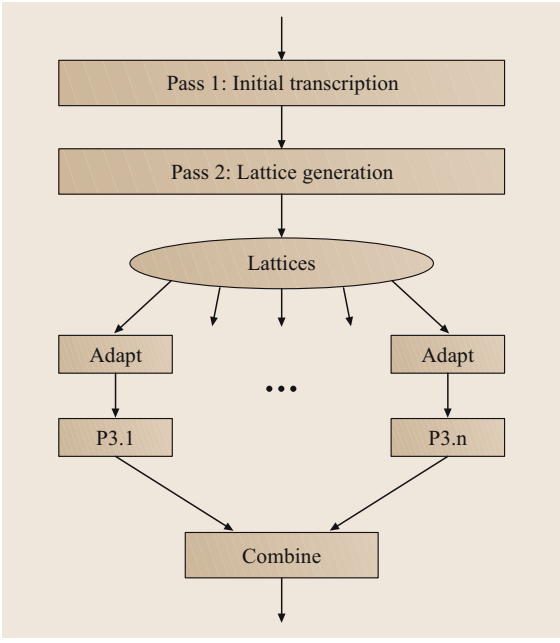


Fig. 27.13 Multipass/system combination architecture

A typical architecture is shown in Fig. 27.13. A first pass is made over the data using a relatively simple SI model set. The 1-best output from this pass is used to perform a first round of adaptation. The adapted models are then used to generate lattices using a basic bigram or trigram word-based language model. Once the lattices have been generated, a range of more-complex models and adaptation techniques can be applied in parallel to provide n candidate output confusion networks from which the best word sequence is extracted. These third pass models may include ML and MPE trained systems, SI and SAT trained models, triphone and quinphone models, lattice-based MLLR, CMLLR, four-gram language models interpolated with class-n-grams and many other variants. For examples of recent large-scale transcription systems see [27.11, 60, 79].

The gains obtained from this type of system combination can vary but overall performance is more robust across a range of task domains. Finally, note that adaptation can work more effectively if the required hypothesized transcriptions are generated by a different system. Thus, cross adaptation is also an increasingly popular architectural option.

27.7 Conclusions

This chapter has reviewed the core architecture of an HMM-based CSR system and outlined the major areas of refinement incorporated into modern systems. Diagonal-covariance continuous-density HMMs are based on the premise that each input frame is independent, and its components are decorrelated and have Gaussian distributions. Since none of these assumptions are true, the various refinements described above can all be viewed as attempts to reduce the impact of these false assumptions. Whilst many of the techniques are quite complex, they are nevertheless effective and overall substantial progress has been made. For example, error rates on the transcription of conversational telephone speech were around 45% in 1995. Today, with the benefit of more

data and the refinements described above, error rates are now well below 20%. Similarly, broadcast news transcription has improved from around 30% word error rate (WER) in 1997 to below 15% today.

Despite their dominance and the continued rate of improvement, many argue that the HMM architecture is fundamentally flawed and performance must asymptote. Of course, this is undeniably true since we have in our own heads an existence proof. However, no good alternative to the HMM has yet been found. In the meantime, the performance asymptote seems still to be some way away.

References

27.1 J.K. Baker: The dragon system – an overview, IEEE Trans. Acoust. Speech Signal Process. **23**(1), 24–29 (1975)

27.2 F. Jelinek: Continuous speech recognition by statistical methods, Proc. IEEE **64**(4), 532–556 (1976)

27.3 B.T. Lowerre: *The Harpy Speech Recognition System, Ph.D. Dissertation* (Carnegie Mellon, Pittsburgh 1976)

27.4 L.R. Rabiner, B.-H. Juang, S.E. Levinson, M.M. Sondhi: Recognition of isolated digits using HMMs with continuous mixture densities, AT&T Tech. J. **64**(6), 1211–1233 (1985)

27.5 L.R. Rabiner: A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE **77**(2), 257–286 (1989)

27.6 P.J. Price, W. Fisher, J. Bernstein, D.S. Pallet: The DARPA 1000-word resource management database for continuous speech recognition, Proc. IEEE ICASSP **1**, 651–654 (1988)

27.7 S.J. Young, L.L. Chase: Speech recognition evaluation: A review of the US CSR and LVCSR programmes, Comput. Speech Lang. **12**(4), 263–279 (1998)

27.8 D.S. Pallet, J.G. Fiscus, J. Garofolo, A. Martin, M. Przybocki: *1998 Broadcast News Benchmark Test Results: English and Non-English Word Error Rate Performance Measures, Tech. Rep.* (National Institute of Standards and Technology, Gaithersburg 1998)

27.9 J.J. Godfrey, E.C. Holliman, J. McDaniel: Switchboard, Proc. IEEE ICASSP **1**, 517–520 (1992)

27.10 G. Evermann, H.Y. Chan, M.J.F. Gales, T. Hain, X. Liu, D. Mrva, L. Wang, P. Woodland: Development of the 2003 CU-HTK Conversational Telephone Speech Transcription System, Proc. IEEE ICASSP (2004)

27.11 H. Soltau, B. Kingsbury, L. Mangu, D. Povey, G. Saon, G. Zweig: The IBM 2004 conversational telephony system for rich transcription, Proc. IEEE ICASSP (2005)

27.12 S.J. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, P. Woodland: *The HTK Book Version 3.4* (Cambridge University, Cambridge 2006), http://htk.eng.cam.ac.uk

27.13 S.J. Young: Large vocabulary continuous speech recognition, IEEE Signal Process. Mag. **13**(5), 45–57 (1996)

27.14 S.B. Davis, P. Mermelstein: Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences, IEEE Trans. Acoust. Speech Signal Process. **28**(4), 357–366 (1980)

27.15 H. Hermansky: Perceptual linear predictive (PLP) analysis of speech, J. Acoust. Soc. Am. **87**(4), 1738–1752 (1990)

27.16 A.P. Dempster, N.M. Laird, D.B. Rubin: Maximum likelihood from incomplete data via the EM algorithm, J. Roy. Stat. Soc. B **39**, 1–38 (1977)

27.17 S.J. Young, J.J. Odell, P.C. Woodland: Tree-based state tying for high accuracy acoustic modelling, Proc. Human Language Technology Workshop (Morgan Kaufman, San Francisco 1994) pp. 307–312

27.18 X. Luo, F. Jelinek: Probabilistic classification of HMM states for large vocabulary, Proc. Int. Conf. Acoust. Speech Signal Process. (1999) pp. 2044–2047

27.19 S.M. Katz: Estimation of probabilities from sparse data for the language model component of a speech recogniser, IEEE Trans. ASSP **35**(3), 400–401 (1987)

27.20 H. Ney, U. Essen, R. Kneser: On structuring probabilistic dependences in stochastic language modelling, Comput. Speech Lang. **8**(1), 1–38 (1994)

27.21 S.F. Chen, J. Goodman: An empirical study of smoothing techniques for language modelling, Comput. Speech Lang. **13**, 359–394 (1999)

27.22 P.F. Brown, V.J. Della Pietra, P.V. de Souza, J.C. Lai, R.L. Mercer: Class-based n-gram models of natural language, Comput. Linguist. **18**(4), 467–479 (1992)

27.23 R. Kneser, H. Ney: Improved clustering techniques for class-based statistical language modelling, Proc. Eurospeech **93**, 973–976 (1993)

27.24 S. Martin, J. Liermann, H. Ney: Algorithms for bigram and trigram word clustering, Proc. Eurospeech **2**, 1253–1256 (1995)

27.25 S.J. Young, N.H. Russell, J.H.S. Thornton: *Token Passing: A Simple Conceptual Model for Connected Speech Recognition Systems, Tech. Rep.*

CUED/F-INFENG/TR38 (Cambridge University, Cambridge 1989)
27.26 K. Demuynck, J. Duchateau, D. van Compernolle: A static lexicon network representation for cross-word context dependent phones, Proc. Eurospeech **97**, 143–146 (1997)
27.27 S.J. Young: Generating multiple solutions from connected word DP recognition algorithms, Proc. IOA Autumn Conf. **6**, 351–354 (1984)
27.28 H. Thompson: Best-first enumeration of paths through a lattice – an active chart parsing solution, Comput. Speech Lang. **4**(3), 263–274 (1990)
27.29 F. Richardson, M. Ostendorf, J.R. Rohlicek: Lattice-based search strategies for large vocabulary recognition, Proc. IEEE ICASSP **1**, 576–579 (1995)
27.30 L. Mangu, E. Brill, A. Stolcke: Finding consensus among words: Lattice-based word error minimisation, Comput. Speech Lang. **14**(4), 373–400 (2000)
27.31 G. Evermann, P.C. Woodland: Posterior probability decoding confidence estimation and system combination, Proc. Speech Transcription Workshop (2000)
27.32 G. Evermann, P.C. Woodland: Large vocabulary decoding and confidence estimation using word posterior probabilities, Proc. IEEE ICASSP (2000) pp. 1655–1658
27.33 V. Goel, S. Kumar, B. Byrne: Segmental minimum bayes-risk ASR voting strategies, Proc. ICSLP (2000)
27.34 J. Fiscus: A post-processing system to yield reduced word error rates: Recogniser output voting error reduction (ROVER), Proc. IEEE ASRU Workshop (1997) pp. 347–352
27.35 D. Hakkani-Tur, F. Bechet, G. Riccardi, G. Tur: Beyond ASR 1-best: Using word confusion networks in spoken language understanding, Comput. Speech Lang. **20**(4), 495–514 (2006)
27.36 J.J. Odell, V. Valtchev, P.C. Woodland, S.J. Young: A one-pass decoder design for large vocabulary recognition, Proc. Human Language Technology Workshop (Morgan Kaufman, San Francisco 1994) pp. 405–410
27.37 X. Aubert, H. Ney: Large vocabulary continuous speech recognition using word graphs, Proc. IEEE ICASSP **1**, 49–52 (1995)
27.38 M. Mohri, F. Pereira, M. Riley: Weighted finite state transducers in speech recognition, Comput. Speech Lang. **16**(1), 69–88 (2002)
27.39 F. Jelinek: A fast sequential decoding algorithm using a stack, IBM J. Res. Dev. **13**, 675–685 (1969)
27.40 D.B. Paul: Algorithms for an optimal A* search and linearizing the search in the stack decoder, Proc. IEEE ICASSP **91**, 693–996 (1991)
27.41 A. Nadas: A decision theoretic formulation of a training problem in speech recognition and a comparison of training by unconditional versus conditional maximum likelihood, IEEE Trans. Acoust. Speech Signal Process. **31**(4), 814–817 (1983)
27.42 B.-H. Juang, S.E. Levinson, M.M. Sondhi: Maximum likelihood estimation for multivariate mixture observations of Markov chains, IEEE Trans. Inform. Theory **32**(2), 307–309 (1986)
27.43 L.R. Bahl, P.F. Brown, P.V. de Souza, R.L. Mercer: Maximum mutual information estimation of hidden Markov model parameters for speech recognition, Proc. IEEE ICASSP **96**, 49–52 (1986)
27.44 V. Valtchev, J.J. Odell, P.C. Woodland, S.J. Young: MMIE training of large vocabulary recognition systems, Speech Commun. **22**, 303–314 (1997)
27.45 R. Schluter, B. Muller, F. Wessel, H. Ney: Interdependence of language models and discriminative training, Proc. IEEE ASRU Workshop (1999) pp. 119–122
27.46 P. Woodland, D. Povey: Large scale discriminative training of hidden Markov models for speech recognition, Comput. Speech Lang. **16**, 25–47 (2002)
27.47 W. Chou, C.H. Lee, B.-H. Juang: Minimum error rate training based on N-best string models, Proc. IEEE ICASSP **93**, 652–655 (1993)
27.48 D. Povey, P. Woodland: Minimum phone error and I-smoothing for improved discriminative training, Proc. IEEE ICASSP **2002**, I–105–I–108 (2002)
27.49 P.S. Gopalakrishnan, D. Kanevsky, A. Nadas, D. Nahamoo, M.A. Picheny: Decoder selection based on cross-entropies, Proc. IEEE ICASSP **1**, 20–23 (1988)
27.50 P.C. Woodland, D. Povey: Large scale discriminative training for speech recognition, ISCA ITRW Automatic Speech Recognition: Challenges for the Millenium (2000) pp. 7–16
27.51 M.J.F. Gales: Semi-tied covariance matrices for hidden Markov models, IEEE Trans. Speech Audio Process. **7**(3), 272–281 (1999)
27.52 A.V. Rosti, M. Gales: Factor analysed hidden Markov models for speech recognition, Comput. Speech Language **18**(2), 181–200 (2004)
27.53 S. Axelrod, R. Gopinath, P. Olsen: Modeling with a subspace constraint on inverse covariance matrices, Proc. ICSLP (2002)
27.54 P. Olsen, R. Gopinath: Modeling inverse covariance matrices by basis expansion, Proc. ICSLP (2002)
27.55 N. Kumar, A.G. Andreou: Heteroscedastic discriminant analysis and reduced rank HMMs for improved speech recognition, Speech Commun. **26**, 283–297 (1998)
27.56 M.J.F. Gales: Maximum likelihood multiple subspace projections for hidden Markov models, IEEE Trans. Speech Audio Process. **10**(2), 37–47 (2002)
27.57 T. Hain, P.C. Woodland, T.R. Niesler, E.W.D. Whittaker: The 1998 HTK system for transcription of conversational telephone speech, Proc. IEEE ICASSP **99**, 57–60 (1999)
27.58 G. Saon, A. Dharanipragada, D. Povey: Feature space Gaussianization, Proc. IEEE ICASSP (2004)
27.59 S.S. Chen, R. Gopinath: Gaussianization, Proc. Neural Information Processing Systems (MIT Press, 2000)

27.60 M.J.F. Gales, B. Jia, X. Liu, K.C. Sim, P. Woodland, K. Yu: Development of the CUHTK 2004 RT04 Mandarin conversational telephone speech transcription system, Proc. IEEE ICASSP (2005)
27.61 L. Lee, R.C. Rose: Speaker normalisation using efficient frequency warping procedures, Proc. IEEE ICASSP (1996)
27.62 J. McDonough, W. Byrne, X. Luo: Speaker normalisation with all pass transforms, Proc. ISCLP, Vol. 98 (1998)
27.63 D.Y. Kim, S. Umesh, M.J.F. Gales, T. Hain, P. Woodland: Using VTLN for broadcast news transcription, Proc. ICSLP (2004)
27.64 J.-L. Gauvain, C.-H. Lee: Maximum a posteriori estimation of multivariate Gaussian mixture observations of Markov chains, IEEE Trans. Speech Audio Process. **2**(2), 291–298 (1994)
27.65 S.M. Ahadi, P.C. Woodland: Combined Bayesian and predictive techniques for rapid speaker adaptation of continuous density hidden Markov models, Comput. Speech Lang. **11**(3), 187–206 (1997)
27.66 K. Shinoda, C.H. Lee: Structural MAP speaker adaptation using hierarchical priors, Proc. ASRU, Vol. 97 (1997)
27.67 C.J. Leggetter, P.C. Woodland: Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models, Comput. Speech Lang. **9**(2), 171–185 (1995)
27.68 M.J.F. Gales: Maximum likelihood linear transformations for HMM-based speech recognition, Comput. Speech Lang. **12**, 75–98 (1998)
27.69 F. Wallhof, D. Willett, G. Rigoll: Frame-discriminative and confidence-driven adaptation for LVCSR, Proc. IEEE ICASSP (2000) pp. 1835–1838
27.70 L. Wang, P. Woodland: Discriminative adaptive training using the MPE criterion, Proc. ASRU (2003)
27.71 S. Tsakalidis, V. Doumpiotis, W.J. Byrne: Discriminative linear transforms for feature normalisation and speaker adaptation in HMM estimation, IEEE Trans. Speech Audio Process. **13**(3), 367–376 (2005)
27.72 P. Woodland, D. Pye, M.J.F. Gales: Iterative unsupervised adaptation using maximum likelihood linear regression, Proc. ICSLP (1996) pp. 1133–1136
27.73 M. Padmanabhan, G. Saon, G. Zweig: Lattice-based unsupervised MLLR for speaker adaptation, Proc. ITRW ASR2000: ASR Challenges for the New Millenium (2000) pp. 128–132
27.74 T.J. Hazen, J. Glass: A comparison of novel techniques for instantaneous speaker adaptation, Proc. Eurospeech **97**, 2047–2050 (1997)
27.75 R. Kuhn, L. Nguyen, J.-C. Junqua, L. Goldwasser, N. Niedzielski, S. Finke, K. Field, M. Contolini: Eigenvoices for speaker adaptation, Proc. ICSLP (1998)
27.76 M.J.F. Gales: Cluster adaptive training of hidden Markov models, IEEE Trans. Speech Audio Process. **8**, 417–428 (2000)
27.77 K. Yu, M.J.F. Gales: Discriminative cluster adaptive training, IEEE Trans. Speech Audio Process. **14**, 1694–1703 (2006)
27.78 T. Anastasakos, J. McDonough, R. Schwartz, J. Makhoul: A compact model for speaker adaptive training, Proc. ICSLP (1996)
27.79 R. Sinha, M.J.F. Gales, D.Y. Kim, X. Liu, K.C. Sim, P.C. Woodland: The CU-HTK Mandarin broadcast news transcription system, Proc. IEEE ICASSP (2006)

31. Natural Language Understanding

S. Roukos

We describe several algorithms for developing natural language understanding (NLU) applications. The algorithms include a rule-based system and several statistical systems. We consider two major types of NLU applications: dialog systems and speech mining. For dialog systems, the NLU function aims to understand the full meaning of a user's request in the context of a human–machine interaction in a narrow domain such as travel reservation. For speech mining applications, the NLU function aims at detecting the presence of a limited set of concepts and some of their relations in unrestricted human–human conversations such as in a call center or an oral history digital library. We describe in more detail a statistical parsing algorithm using decision trees for dialog systems and two word-tagging algorithms for speech mining.

Human language is a system for translating thought into physical output (spoken or written) that enables humans to communicate. Natural language understanding (NLU) covers the area of building systems that recover the meaning of spoken or written utterances, thereby enabling human–machine communication. We define NLU as a transduction from text (including text obtained by automatic speech recognition systems) into a formal semantic representation. The formal representation of meaning (or semantics) has been a challenging problem. Crucially, to mitigate the semantic representation challenge, the domain of discourse in NLU applications is in limited tasks/domains (e.g., flight reservations) to allow for building a comprehensive model of the semantics that need to be modeled by the system to enable effective human–machine interaction. [The Defense Advanced Research Projects Agency (DARPA) has funded pioneering research in the air travel information system (ATIS) project in the early 1990s and a follow-on effort on dialog systems for travel reservation called the DARPA Communicator in the 1998–2002 period.]

The communication situation is not symmetric in human–machine dialog systems: the speaker, who may or may not have a good model of the system's capabilities, has a large range of semantic actions while the system has a much narrower and application-specific repertoire and definitely does not have a *good* model of the speaker's semantic repertoire. However, NLU has been used effectively to build dialog systems that are increasingly replacing human operators in call centers by improving the range of tasks that can be handled without the need of a call-center agent. The effectiveness of these solutions is measured by the automation rate, the fraction of calls that are fully handled by the dialog system; for many call-center applications, an increase of a few percentage points in the automation rate typically justifies a multimillion-dollar NLU development project. In this chapter, we will focus on spoken language systems, which have provided the most fertile grounds for recent research on NLU, even though there has been a few web-based chat interfaces powered by NLU technology that we will not discuss further.

31.1 Overview of NLU Applications

The art of building NLU systems and spoken dialog systems (the latter to be discussed in more detail in Chap. 35) is in defining a good representation of the semantics of the application that can be used to map a user's utterance into the application semantics. We describe several example applications to illustrate the range of complexity of the required semantics for current NLU systems.

- Password reset: the user calls a system to reset his password for a specific software application. The system prompts the user to give either his serial number or full name; once the system identifies the user, it asks for confirmation for resetting the password; once the user confirms by speaking phrases such as *yes*, *sure*, or *I guess so*, it resets the password. This application and other similar ones such as package tracking (by users providing a tracking number to the system) are fairly simple in the sense that an utterance has a simple meaning. For password reset, the semantics are a list of user names and the *Yes*/*No* values to confirmation questions.
- Call routing: the user may be asked *How may I help you* and the user response is classified into one of many classes (typically tens of classes) that are used to direct the call to various departments (or call-center operators) in a large business. For example, if the user says *I want to check my credit-card balance* the user call is transferred to the credit-card department.
- Mutual funds: the user calls the system to find the closing price of a mutual fund, find balances in his account, purchase or sell shares in a mutual fund, transfer funds from one account to another, and to get a list of the most recent transactions in his account. The mutual fund application is one of many applications in the financial domain that have been deployed such as stock trading and banking.
- Flight reservation: the user calls the system to find available flights, determine the best options, the price, make a reservation, and optionally make car and hotel reservations. This and the previous application represent the prototypical NLU applications that use natural language parsing technology to extract the meaning of a sentence.
- Speech-to-speech (S2S) translation in narrow domains (e.g., medical and travel domains): in the medical domain, the system mediates a conversation between a patient speaking one language and a doctor who speaks another language. The patient's speech is recognized and *understood* by filling in a template such as type of symptom, time of onset of symptom, etc. These templates represent the type of information that the system can translate between the two languages. The completed template is then used to generate a natural language sentence in the target language that is then synthesized to the doctor. A similar template is completed for the doctor's utterances. These speech-to-speech systems, many of which are currently being developed under DARPA's spoken language communication and translation system for tactical use (TRANSTAC) program, rely heavily on the narrow domain of semantics to achieve reasonable performance. (These speech-to-speech systems are to be contrasted with broad-domain S2S systems that rely on broad-domain machine translation from a source language to a target language as is being developed in the European Union TC-Star and the DARPA global autonomous language exploitation (GALE) projects.)
- Speech mining: these are information extraction systems that process human-to-human calls in a call center using large-vocabulary speech-to-text followed by information extraction systems that identify topics and/or specific concepts in customer calls. For example, the systems can monitor if an agent has been following guidelines on thanking customers at the end of transactions. Speech mining solutions are still in their infancy but they are starting to be used in call centers. They rely on new understanding technology based on the information extraction technology; we will give a brief overview of information extraction later in this chapter.

Typically, applications such as password reset and call routing are not referred to as NLU applications since they require a simple mapping of phrases to atomic semantic concepts; the former is a simple map of words and phrases to class labels while the latter uses classifiers that classify utterances using word and phrase features into one of N classes. The term NLU is reserved for those applications that have a much larger range of semantics such as the mutual funds, travel, and S2S medical applications described above. In these typical NLU applications, each utterance may have multiple *concepts* that may be combined into a more-complex request. Natural language parsing technology is used to under-

stand the structure of users' requests. We will describe in more detail how to build statistical parsers to extract the meaning of an utterance. Finally, the information extraction applications for speech mining typically work on much longer user utterances (as expected when one is analyzing human–human telephone calls) and extract more local pieces of information. These applications use information extraction technology that does not attempt to understand the whole utterance but rather *smaller* pieces of it. We describe in Sect. 31.2 statistical parsing and in Sect. 31.4 statistical models for information extraction.

31.1.1 Context Dependence

The meaning of a speech utterance may depend on the context of the conversation between the user and the machine. For context-independent sentences, the meaning is completely specified by the speech utterance independently of the conversation context. For example, the meaning of the user's utterance, *give me the first nonstop flight from Boston to Austin on Wednesday March 3rd*, is wholly extracted from the actual spoken sentence without regard to context, while the meaning of fragments such as *the second one* or *do you have an earlier flight* or the one word utterance *Chicago* depends on the context of the conversation; in these latter cases, the meaning depends on what the system said to the user in the previous turn of the conversation. *Chicago* will be the *from* or *to* city depending on what the system asked earlier. One approach is to define an ambiguous meaning representation for these utterances that another component, typically called a dialog manager, would disambiguate using the conversational state of the system. Another approach is to provide the statistical parser with some features that describe the state of the conversation; this context can be used to identify the meaning of the next fragment. We will illustrate one example of the latter approach in Sect. 31.2.

31.1.2 Semantic Representation

Since we need to map a user's utterance to its meaning using the semantic model of the application, a key step in building an NLU system is to design the semantic language for representing the meaning of the user's utterances. There are many different representations, we discuss one example. This formal language for the possible semantics of the ATIS travel task is based on the NL-MENU language developed by Texas Instruments for the ATIS task. This is a canonical example that exemplifies NLU interfaces to relational database management systems (RDBMS). The formal language is somewhat related to the structured query language (SQL) formal language to RDBMS with specific names that are specified by the various fields in the tables of the application's RDBMS. For example, the NL-MENU language used in the ATIS task has the following example formal *sentences* (bold words are keywords, upper-case words are variables):

- **List Flights from_city** CITY **to_city** CITY
- **List Flights from_city** CITY **to_city** CITY **flying_on** DATE
- **List Flights from_city** CITY **to_city** CITY **flying_on** DATE **departing_after** TIME

The formal language has its own interpreter to map to actual application programming interface (API) calls of back-end systems (that provide flight information or mutual fund account information). In the case of NL-MENU the translation to SQL is straightforward. In some applications, there have been efforts at using predicate logic (in particular, first-order logic) to represent the application semantics but these have not been commercially successful.

In general, application developers define a formal language for the semantics that the backend system can support. The design of the formal language has to take into account the fact that it will be used to interpret the users' natural language sentences. Corresponding to the second formal expression above, the user may have said any one of the following utterances:

- I want to go from Boston to Austin tomorrow;
- Boston to Austin this Tuesday;
- What flights do you have that go to Austin; I am planning to leave from Boston next Saturday.

Therefore, the two ingredients for an NLU system are

- a formal language specification of the application semantics
- a system that maps natural language utterances to their corresponding semantic representation

This latter step is the NLU or interpretation step, and is the main focus of this chapter, where we will discuss how to build the NLU component.

31.2 Natural Language Parsing

Table 31.1 Grammar for understanding sentences in the mutual fund domain

S[$1.$2.$3]	→	INTRO? ACTION [$1] NUMBER [$2] of FUND [$3]
INTRO	→	I would like to | I want to | Please
ACTION [$1]	→	get me [BUY] | buy [BUY] | sell [SELL]
NUMBER [$1]	→	(one thousand [1000] |one hundred [100] |ten [10]) shares of?
FUND [$1]	→	Vanguard 500 index [VFINX] | Fidelity contrafund [FCNTX]

Early approaches to understand a sentence were based on developing grammars to translate a sentence into its meaning. Table 31.1 shows a canonical example using finite-state transducers (with Backus–Nauer form (BNF) notation to define the grammar). We use upper-case words to denote variables (also known as nonterminals) that are the left-hand sides, which are rewritten as the right-hand side (RHS), where the vertical line (|) denotes rewriting alternatives; elements followed by a question mark are optional (zero or one occurrence); the elements between brackets are the output language of the transducer. $1 denotes the first variable in the RHS for the output language, $2 will denote the second output variable, and so on.

In the above sample grammar, the first rule indicates concatenation of the three variables that represent the meaning of the three RHS variables. The third rule states that the meaning of ACTION is the meaning of the matching right-hand side, either $1=BUY or $1=SELL depending on which matches the input. A valid sentence that can be parsed by this grammar is: *Please get me one thousand shares of Fidelity Contrafund*, which would be translated to *BUY.1000.FCNTX*, which can be used to execute the request in a backend system.

In these transducers, the grammar developer has to capture both the range of input expressions of users and their corresponding output meaning. The above finite-state transducer is an example of the BNF-grammar-based parsers that can be developed. Other formalisms for grammar development have been pursued. However, these manual approaches to grammar development have since been replaced by statistical-based approaches. Instead of writing rules to cover the interpretation of input sentences, the statistical approach relies on the creation of a manually annotated corpus, called a treebank, of sentences, which are parsed by hand with their semantic meaning. The corpus is created by asking users to give example sentences on how they would talk to the system; the process is refined by actually collecting data in a pilot phase of the project. This corpus is then manually parsed. Figure 31.3 shows an example semantic parse tree of a sentence. To build a statistical parser, a corpus of 1000–10 000 sentences that are manually parsed is created and is used as a training corpus to build the parser. We now describe one method for developing a statistical parser based on decision trees.

Before we describe parser development, we need to discuss how the semantics of a sentence are captured in its parse tree. One has significant freedom in specifying the parsing style. Figures 31.1 and 31.2 give two options for the definition of an appropriate parse tree. Notice that these semantic parses may not agree with traditional syntactic structures such as those found in earlier *treebanks* [31.1]. Earlier attempts at statistical models for NLU relied on segmenting the sentence as a sequence of hidden Markov model (HMM)

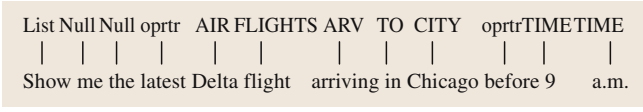


Fig. 31.1 Example semantic labeling

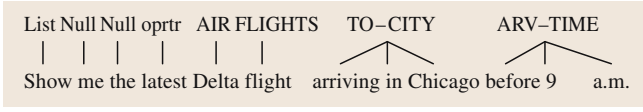


Fig. 31.2 Example semantic parse tree

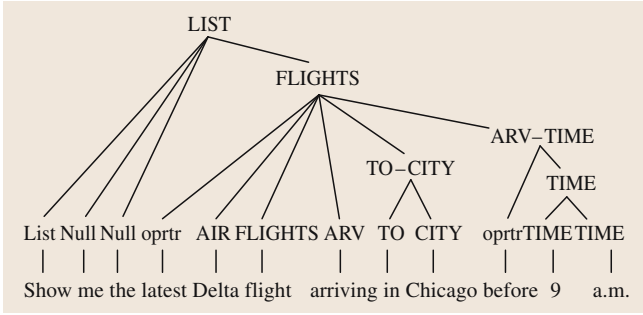


Fig. 31.3 Semantic parse tree in the travel domain

states [31.2] (called *concepts*) and best illustrated in Fig. 31.2. This segmentation-based approach is reminiscent of the information extraction [31.3] approach to language understanding described in more detail in Sect. 31.3. We believe that more-effective models can be achieved with the hierarchical structure that a parse tree can produce for NLU applications. There have been attempts to use HMM models to capture recursive structure or nesting of concepts by designing a state space where state transitions allow one to form a hierarchy of states (and the corresponding transitions) as in the work on hidden understanding models [31.4], but a parsing-based solution is much more natural and effective.

It is a natural tendency to try to capture very detailed semantics when one is designing the guidelines for semantic annotation (i. e., parsing) for a new application. But we have found that the richness of the parse tree should be simplified for two reasons.

- To identify the minimal constituents needed to enable more-accurate models for parsing. We have found that highly nested structures tend to lead to lower parsing accuracy; while this is somewhat of an art, the nesting should be kept to the minimum needed to capture the semantic richness of the domain.
- The flatter the structure the easier it is to explain the annotation guidelines to human annotators and the faster they can do the manual parsing accurately and consistently. Consistency is an important requirement for building real systems since these annotations are application-specific, and consistent treebanks are needed to learn an accurate statistical parser.

Another set of understanding models are based on statistical translation models. In particular, the maximum-entropy (MaxEnt)-based approach described in [31.5] achieves very good results but requires a relatively large training corpus. The method also requires a comprehensive list of all possible formal language requests in the domain. While this method can be used when one has large amounts of data, it is more difficult to bootstrap a system with it and hence it has limited applicability in deploying new NLU applications.

31.2.1 Decision Tree Parsers

Some approaches to statistical parsing use a grammar, possibly extracted from a treebank, and a probabilistic model such as a probabilistic context-free grammar (PCFG) to determine the most probable parse for a sentence. In this work, we describe an approach that does not use a grammar to construct a probabilistic model $p(T|w_1^n)$, where T is the parse tree of the word sequence w_1^n. We present a method for constructing a model for the conditional distribution of trees given a sentence for all possible trees for that sentence. To parse, we need to find the most probable tree given the model:

$$T^* = \underset{T \in \mathcal{T}(w_1^n)}{\operatorname{argmax}}\ p(T|w_1^n)\,, \tag{31.1}$$

where the maximization is over all trees that span the n-word sentence.

The probabilistic models we explore depend on the *derivational* order of the parse tree. In [31.6], this type of model is referred to as a *history-based* grammar model where a (deterministic) leftmost derivation order is used to factor the probabilistic model. In this work, we use a set of bottom-up derivations of parse trees. Traditionally the derivation order identifies the order of application of grammar rules, i. e., it corresponds to the order of node expansion in a parse tree; in this work, we extend the notion to identify the order in which edges in a tree are created, which includes partial constituents or nodes. In [31.7], both a hidden derivational model as well as a deterministic derivational model are explored to assign the probability of a parse tree. In this work, we only discuss the deterministic bottom-up leftmost (BULM) derivation order.

We now discuss the derivation history model, the parsing model, the probabilistic models for node features, and the training algorithm.

Treebanks are a collection of n-ary branching trees, where each node in a tree is labeled by either a nonterminal label (also known as a constituent) or a pre-terminal label (called a tag) obtained by tagging a word (also known as a terminal). If a parse tree is interpreted as a geometric pattern, a constituent is no more than a set of edges that meet at the same node, each of which has a label. In Fig. 31.3, the nonterminal *ARV-TIME*, which spans the tags *oprtr time time* corresponding to the words *before 9 a.m.*, consists of an edge extending to the right from *oprtr* and an edge extending from the nonterminal *TIME* to the left.

We introduce a new definition for a derivation of a parse tree by using Fig. 31.4, which gives a partial subtree used in our parser. We associate with every node in the parse tree two features: a name which is either a tag or a no-terminal label, and an extension, which indicates whether the edge going to its parent is going right, left, up, or is unary. The unary result corresponds to a renaming of a nonterminal. By specifying the two

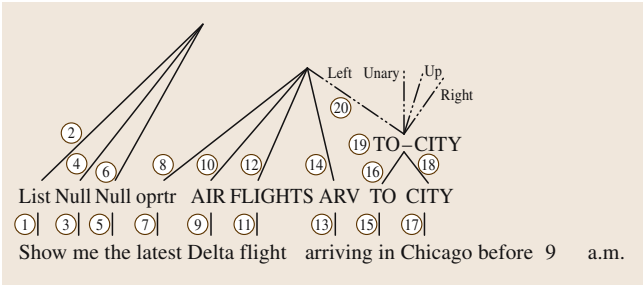


Fig. 31.4 Partial semantic parse tree with bottom-up leftmost derivation

features (name and extension) for each node we can reconstruct the parse tree. The order of the nodes in which we specify these two features defines the derivation order. We only consider bottom-up leftmost derivations. In a BULM derivation, a node is named first; it may be extended only after it is named, and it is not named until all of the nodes beneath it are extended. When naming a node we refer to the derivation steps as either tagging (when the node is a pre-terminal) or labeling (when the node corresponds to a nonterminal).

We define a derivation as the sequence of actions needed to generate the parser tree. The actions come in three flavors: tagging, extending, and labeling. We illustrate these three action types by defining the derivation of the parse subtree shown in Fig. 31.4. We start by tagging the first word in a sentence; this is step 1. We then extend right (denoted as step 2), then we do step 3, which is to tag the second word as 'Null'. After step 19, where we have labeled the node $TO-CITY$, we now have four possible extension actions: left, unary, up, right, each with a probability that depends on the partial parse derivation so far. This corresponds to step 20, denoted by derivation step d_{20}, where node 19 is the active node.

At each decision point, there is an active node where the next action is performed. So the probability of a parse tree T with derivation d is:

$$p(T|w_1^n) = \prod_{1<j<|d|} p(d_j|d_1^{j-1}) \,, \tag{31.2}$$

where the probability of taking action d_j given the previous derivation steps and the input sentence w_1^n can use features from the partial *parse* constructed so far. In [31.7], a hidden derivational model is described that has a slightly better accuracy than the single derivation model discussed here; but in real-world deployments the speed benefit of single derivation models outweighs the small decrease in accuracy.

One can use a number of statistical models for the incremental action model, ranging from MaxEnt models such as that described in [31.8] or the linear models as described by [31.9]. However, we will describe a model using decision trees since it has been used with a commercial offering called IBM ViaVoice Telephony for building telephony-based NLU systems. We essentially have three models:

- Tag model: the tag feature value prediction is conditioned on a window of two words to the left, two words to the right, and two nodes to the left (the two nodes to the right are the same as the two words to the right in a BULM derivation). So if we denote the active node N_k to tag word w_i, the two nodes to the left would be denoted by N_k^{-2} and N_k^{-1}. We have:

  $$p(t_i|\text{context}) \approx p(t_i|w_{i-2}^{i+2}, t_{i-2}^{i-1}, N_k^{-2}, N_k^{-1}) \,. \tag{31.3}$$

  Note that to tag a word such as in step 15, the word *in*, the two nodes to the left are the partially created nodes that span *latest delta flight arriving* for N_{15}^{-1} and *Show me the* for N_{15}^{-2}, respectively. The two words to the right are *Chicago* and *before*. The decision tree can use binary features (questions) that explore the substructure of the nodes. The words are represented by a binary bit vector that indicates membership of M classes. Labels of nodes are also represented by a bit vector. These bit vectors define the binary questions used in building a decision tree model for the tagging decision.
- Extension model: similarly to the tag model, the extension model uses a window of two nodes to the left and two nodes (words in BULM) to the right and the current node N_k:

  $$p(e_k|\text{context}) \approx p(e_k|N_k^{-2}, N_k^{-1}, N_k, w_k^{+1}, w_k^{+2}) \,, \tag{31.4}$$

  where w_k^{+i} denotes the i-th word to the right of the span of the current node N_k. A number of features are used to represent a node N: the label of the node, its rightmost child, its leftmost child, the rightmost word in its span, and the leftmost word. Each of these are represented by a bit vector of class memberships. One set of classes and corresponding bit vectors can be created by word clustering [31.10].
- Label model: we also use a window of size five centered on the node to label N_k. The model is given by:

  $$p(L_k|\text{context}) \approx p(L_k|Q_k, N_k^{-2}, N_k^{-1}, w_k^{+1}, w_k^{+2}) \,, \tag{31.5}$$

  where Q_k is a set of questions that are defined, and explore the subtree under node N_k.

Statistical Decision Trees

These probability distributions are modeled by traditional decision trees with binary questions on the history. These binary questions rely heavily on defining classes by clustering words, tags, labels, or extension names. The clustering algorithms are described in [31.10]. One can also add head propagation rules as described in [31.7] to enrich the features used by the decision tree.

31.3 Practical Implementation

We have found that, the shallower the tree, the easier it is to build decision tree parsers with high accuracy. One approach to achieving shallower trees is to use a two-pass approach to semantic parsing.

31.3.1 Classing

For this first pass of semantic parsing the input sentence is marked by finding chunks. For example in the travel domain, time, location, and date expressions are marked. Figure 31.5 shows an example sentence. These trees are quite shallow and reminiscent of shallow parsing. The classed sentence becomes *Show me the latest AIR flight arriving in CITY before TIME.* This sentence is then passed to the second-pass parser.

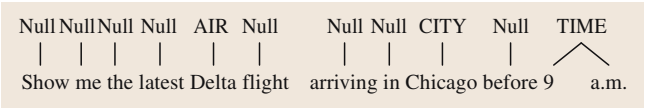


Fig. 31.5 Classer output

31.3.2 Labeling

Once a sentence is tagged we replace the chunks by their classes and the resulting sentence is parsed for more-detailed semantics. The sentence in Fig. 31.5 becomes *Show me the latest AIR flight arriving in CITY before TIME*, which is now the input to the second-pass parser.

This cascade model for semantic parsing has been found to yield more-robust parsers for multiple NLU applications.

There has been a large amount of work on developing statistical parsers [31.11–14], but these have mostly been used with the Penn treebank for syntactic parsing and have a rather large training set of about 1 million words of parsed data. They have not been evaluated for NLU application where the available training sets are rather small, typically ranging from 10 000 thousand to 100 000 words of parsed data.

For practical NLU application development, a recipe for building a parser depends on the availability of sample sentences. This can be initiated by asking people familiar with the system design to write down 10–50 sentences each to obtain a starting set that can be used to bootstrap a first system. The system is then used to produce another round of data collection with employees or paid volunteers to get a few thousand sentences, which is used to build a more-robust semantic parser that can then be used in a pilot deployment, which after a one- or two-month pilot phase can become the production system. Unfortunately, this iterative cycle of data collection and system building is necessary for dialog applications since it is hard to anticipate what users will say when they interact with these application-specific systems. Also in this iterative approach, one can run the parse annotation tool in correction mode by parsing the next batch of data with the current parser and asking human annotators to correct the machine-produced parses. This correction mode speeds up the annotation process significantly. There has also been some work [31.15] on exploring active learning methods to select which sentences humans should manually annotate to reduce the annotation effort.

To handle context dependence, the parser can examine the input sentence and additional input that describes context, such as keywords summarizing the prompt that the system delivered to the user or other state variables. The context is used in the decision tree as additional features to disambiguate the parse of an input sentence.

31.4 Speech Mining

Another class of applications for natural language understanding is for systems that analyze human–human dialogs. These are referred to as speech mining applications and are starting to be deployed in call centers.

Speech mining covers a broad range of applications from early discovery of trends in a call center to monitoring and training call-center representatives (CSRs). In addition, speech mining of other types of conversations such as in broadcast news, talk shows, and oral history archives are currently a research focus.

In these applications the range of concepts discussed in the conversations is typically much larger than the dialog applications we have discussed so far. So instead of building a system that understands the meaning of all that is discussed, these speech mining applications focus on a specific subset of concepts and determine whether these concepts occur in the conversation. These information extraction systems can be viewed as word-tagging systems where concepts are spotted in the running speech (or text) and that possibly also detect relations between the spotted concepts. We give two example applications.

- *Speech mining for call centers:* one subset of application is for monitoring and training CSRs in a call center; the system is set up to detect that certain phrases are used by the CSR, such as did the CSR use the appropriate phrases in the closing script (e.g., *Thank you for calling* or *Anything else*) or for incipient problem detection by finding certain phrases that indicate a problem and its type from a customer's comments. The shallow semantics depend on the application but they may be as simple as the following two concepts indicated in the following customer's utterance: *I do not know how to install an additional [HARDWARE hard disk] I [REQUEST need your help];* basically this is about a hard disk and the customer is asking for support.

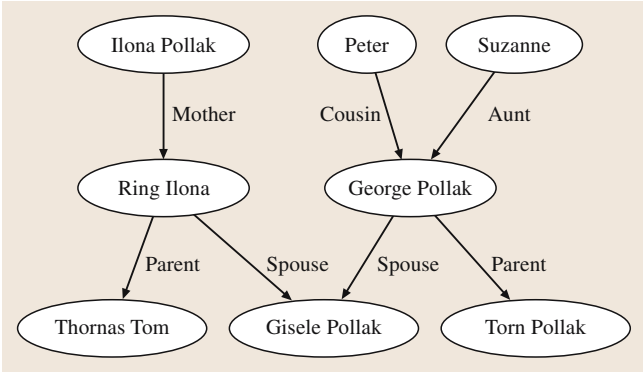


Fig. 31.6 Personal network extracted from human-transcribed interview

- *Speech mining for discovery:* a significant number of audio databases are becoming available and searchable. Examples are oral history collections such as the collection used in the multilingual access to large spoken archieves (MALACH) project [31.16], which consists of a collection of two-hour interviews with holocaust survivors (there are 50 000 interviews in about 30 languages). Information extraction can be used to analyze these recordings to find mentions of people and find relationships between the people mentioned. One example is to extract all the people that are part of the extended family that are mentioned in the interview by doing entity extraction and *family* relation extraction between the spotted entities. Figure 31.6 shows an example network extracted from human-transcribed speech, and Fig. 31.7 shows the corresponding reference network. In Fig. 31.6, we see entities extracted and relations between them such as: *George* is the spouse of *Gisele*. We also notice errors due to incorrect entity extraction and relation extraction such as *Ring Iona* is a spouse of *Gisele Pollack*.

31.4.1 Word Tagging

We now describe two approaches for tagging words with semantics concepts: a linear classifier – the robust risk minimization (RRM) classifier – and a log-linear classifier – the maximum-entropy (MaxEnt) classifier. Both methods can integrate arbitrary types of information and make a classification decision by aggregating all information available for a given classification.

Let $\mathcal{C} = \{c_1, \ldots, c_n\}$ be the set of predicted classes, and $\mathcal{X}$ be the example space. Each example $x \in \mathcal{X}$ is associated with a vector of binary features $f(x) = (f_1(x), \ldots, f_m(x))$. We also assume the existence of a training data set $\mathcal{T} \subset \mathcal{X}$ and a test set $\mathcal{E} \subset \mathcal{X}$.

The RRM algorithm [31.9] constructs n linear classifiers $(C_i)_{i=1,\ldots,n}$ (one for each predicted class), each predicting whether the current example belongs to the class or not. Every such classifier C_i has an associated feature weight vector, $(w_{ij})_{j=1,\ldots,m}$, which is learnt during the training phase so as to minimize the classification error rate.

At test time, for each example $x \in \mathcal{E}$, the model computes a score

$$a_i(x) = \sum_{j=1}^{m} w_{ij} \cdot f_j(x) \,,$$

and labels the example with either the class corresponding to the classifier with the highest score, if above 0, or 'outside', otherwise. The RRM classifier is:

foreach $x \in \mathcal{E}$
  foreach $i = 1 \ldots n$
    $a_i[x] = \sum_{j=1}^{m} w_{ij} \cdot f_j(x)$
  $\text{class}(x) \leftarrow \arg \max_i a_i[x]$.

This linear classifier is used for *sequence classification* by converting the activation scores into probabilities (through the *soft-max* function, for instance) and using the standard dynamic programming search algorithm (also known as a Viterbi search) to find the most likely tag sequence for the word sequence.

Somewhat similarly, the MaxEnt algorithm has an associated set of weights $(\alpha_{ij})_{j=1,\ldots,m}^{i=1,\ldots,n}$, which are estimated during the training phase so as to maximize the likelihood of the data [31.8]. Given these weights, the model computes the probability distribution of a particular example x as follows:

$$P(c_i|x) = \frac{1}{Z} \prod_{j=1}^{m} \alpha_{ij}^{f_j(x)} , \quad Z = \sum_i \prod_j \alpha_{ij}^{f_j(x)} ,$$

where Z is a normalization factor.

After computing the class probability distribution, the assigned class is the most probable one a posteriori. A schematic for the application of the MaxEnt approach to the test data is:

foreach $x \in \mathcal{E}$
  $Z \leftarrow 0$
  foreach $i = 1 \ldots n$
    $p_i[x] = \prod_{j=1}^{m} \alpha_{ij}^{f_j(x)}$
  Normalize(p)
  $\text{class}(x) \leftarrow \arg \max_i p_i[x]$.

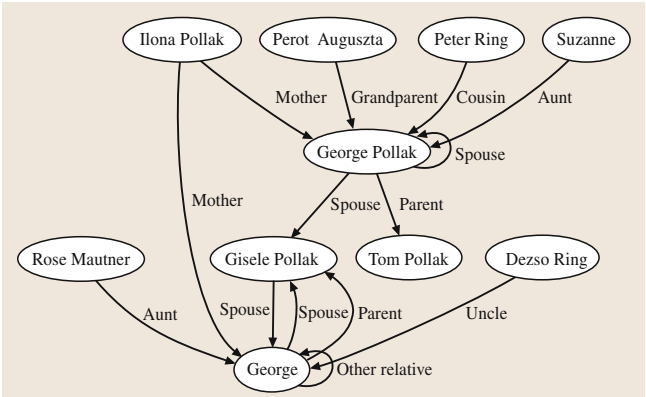


Fig. 31.7 Correct personal network extracted from human-transcribed interview

Similarly to the RRM model, we use the model to perform sequence classification, through dynamic programming. Within this framework, any type of feature can be used, enabling the system designer to experiment with interesting feature types, rather than worry about specific feature interactions. We have found both methods to be equally as effective in a number of situations. The selection of features depends on the problem and the available linguistic resources for the language at hand. Example features are the words in a five-word window, the stems of these words, and prefixes and suffixes of various lengths. Also, one can include part-of-speech tags and WordNet synsets [31.17], if available for the language of interest. Domain-specific lexicons such as a list of product names can also be useful.

One uses the above classifiers with a Viterbi search to build a mention detection system for all the entities/concepts of interest. Following mention detection, we run a second classifier to find relations between concepts or every pair of mentions within a unit (e.g., a user turn). Similarly, the features used to detect whether a particular relation exist uses the type of mentions, the words between the mentions, and the parse tree of the sentence. A more-detailed description of one particular relation extraction system is given in [31.18].

31.5 Conclusion

Natural language understanding systems are increasingly being used in telephony-based dialog systems, as speech recognition systems that support large-vocabulary n-gram language models become more accurate. The ability to handle freeform user requests enables the automation of more-complex applications, resulting in significant cost reduction in call centers. The statistical modeling approach described in this chapter has proven

to be effective for high-accuracy systems. In addition, it enables an efficient solution for deploying an application in many languages. By using the same semantics and translating the training data from a system developed in one language, one can efficiently create equivalent systems in other languages. A bilingual system that allows a speaker to speak in either of two languages on a turn-by-turn basis can also be built. These bilingual systems may prove useful for many customer segments. In addition, these data-driven methods allow continuous tuning of a deployed system as more usage data is acquired in order to achieve higher levels of accuracy.

References

31.1 M.P. Marcus, B. Santorini, M.A. Marcinkiewicz: Building a large annotated corpus of English: The Penn Treebank, Comput. Linguist. **19**, 313–330 (1993)

31.2 E. Levin, R. Pieraccini: CHRONUS: *The Next Generation, ARPA Spoken Language Systems Technology Workshop* (1995) pp. 269–271

31.3 R. Florian, H. Hassan, A. Ittycheriah, H. Jing, N. Kambhatla, X. Luo, N. Nicolov, S Roukos: A statistical model for multilingual entity detection and tracking, Proc. of HLT-NAACL 2004 (2004) pp. 1–8

31.4 R. Schwartz, S. Miller, D. Stallard, J. Makhoul: Language understanding using hidden understanding models, Proc. of ICSLP-96 (1996)

31.5 K. A. Papineni, S. Roukos, R. T. Ward: Feature-based language understanding, Proc. of the 5th European Conference on Speech Communication and Technology (1997)

31.6 E. Black, F. Jelinek, J. Lafferty, D. M. Magerman, R. Mercer, S. Roukos: Towards history-based grammars: Using richer models for probabilistic parsing, Proc. of the Association for Computational Linguistics (1993) pp. 31–37

31.7 F. Jelinek, J. Lafferty, D. M. Magerman, R. Mercer, A. Ratnaparkhi, S. Roukos: Decision tree parsing using a hidden derivation model, Proc. of the 1994 Human Language Technology Workshop (1994) pp. 272–277

31.8 A. Berger, S. Della Pietra, V. Della Pietra: A maximum entropy approach to natural language processing, Comput. Linguist. **22**(1), 39–71 (1996)

31.9 T. Zhang, F. Damerau, D.E. Johnson: Text chunking based on a generalization of winnow, J. Machine Learning Res. **2**, 615–663 (2002)

31.10 P. F. Brown, V. J. Della Pietra, P. V. deSouza, J. C. Lai, R. L. Mercer: Class-based *n*-gram models of natural language, Proc. of the IBM Natural Language ITL (1990)

31.11 A. Ratnaparkhi: A linear observed time statistical parser based on maximum entropy models, Proc. of the Second Conference On Empirical Methods in Natural Language Processing (EMNLP-2) (1997)

31.12 M. Collins: Head-driven statistical models for natural language parsing, Comput. Linguist. **29**, 589–637 (2003)

31.13 D. M. Bikel: On the parameter space of generative lexicalized statistical parsing models, PhD Dissertation, University of Pennsylvania (2004)

31.14 E. Charniak, M. Johnson: Coarse-to-fine n-best parsing and MaxEnt discriminative reranking, Proc. of Association for Computational Linguistics ACL (2005) pp. 173–180

31.15 M. Tang, X. Luo, S. Roukos: Active learning for statistical natural language parsing, Proc. of the Association for Computational Linguistics (2002)

31.16 D. Oard, D. Soergel, D. Doermann, X. Huang, G. C. Murray, J. Wang, B. Ramabhadran, M. Franz, S. Gustman, J. Mayfield, L. Kharevych, S. Strassel: Building an information retrieval test collection for spontaneous conversational speech, Proc. of SIGIR'04 (2004)

31.17 G. A. Miller: WordNet: A lexical database, Commun. ACM, 38(11) (1995)

31.18 N. Kambhatla: Minority vote: At-least-N voting improves recall for extracting relations, Proc. of COLING/ACL 2006 (2006) pp. 460–466

28. Speech Recognition with Weighted Finite-State Transducers

M. Mohri, F. Pereira, M. Riley

This chapter describes a general representation and algorithmic framework for speech recognition based on weighted finite-state transducers. These transducers provide a common and natural representation for major components of speech recognition systems, including hidden Markov models (HMMs), context-dependency models, pronunciation dictionaries, statistical grammars, and word or phone lattices. General algorithms for building and optimizing transducer models are presented, including composition for combining models, weighted determinization and minimization for optimizing time and space requirements, and a weight pushing algorithm for redistributing transition weights optimally for speech recognition. The application of these methods to large-vocabulary recognition tasks is explained in detail, and experimental results are given, in particular for the North American Business News (NAB) task, in which these methods were used to combine HMMs, full cross-word triphones, a lexicon of 40 000 words, and a large trigram grammar into a single weighted transducer that is only somewhat larger than the trigram word grammar and that runs NAB in real time on a very simple decoder. Another example demonstrates that the same methods can be used to optimize lattices for second-pass recognition.

28.1 Definitions

Much of current large-vocabulary speech recognition is based on models such as hidden Markov models (HMMs), lexicons, or n-gram statistical language models that can be represented by *weighted finite-state transducers*. Even when richer models are used, for instance context-free grammars for spoken-dialog applications, they are often restricted, for efficiency reasons, to regular subsets, either by design or by approximation [28.1–3].

A finite-state transducer is a finite automaton whose state transitions are labeled with both input and output symbols. Therefore, a path through the transducer encodes a mapping from an input symbol sequence, or *string*, to an output string. A *weighted* transducer puts weights on transitions in addition to the input and output symbols. Weights may encode probabilities, durations, penalties, or any other quantity that accumulates along paths to compute the overall weight of mapping an input string to an output string. Weighted transducers are thus a natural choice to represent the probabilistic finite-state models prevalent in speech processing.

We present a detailed view of the use of weighted finite-state transducers in speech recognition [28.4–10]. We show that common methods for combining and optimizing probabilistic models in speech processing can be generalized and efficiently implemented by translation to mathematically well-defined operations on weighted transducers. Furthermore, new optimization

opportunities arise from viewing all symbolic levels of speech recognition modeling as weighted transducers. Thus, weighted finite-state transducers define a common framework with shared algorithms for the representation and use of the models in speech recognition that has important algorithmic and software engineering benefits.

We begin with an overview in Sect. 28.2, which informally introduces weighted finite-state transducers and algorithms, and motivates the methods by showing how they are applied to speech recognition. This section may suffice for those only interested in a brief tour of these methods. In the subsequent two sections, we give a more-detailed and precise account. Section 28.3 gives formal definitions of weighted finite-state transducer concepts and corresponding algorithm descriptions. Section 28.4 gives a detailed description of how to apply these methods to large-vocabulary speech recognition and shows performance results. These sections are appropriate for those who wish to understand the algorithms more fully or wish to replicate the results.

28.2 Overview

We start with an informal overview of weighted automata and transducers, outlines of some of the key algorithms that support the ASR applications described in this chapter – *composition*, *determinization*, and *minimization* – and their application to speech recognition.

28.2.1 Weighted Acceptors

Weighted finite automata (or weighted acceptors) are used widely in automatic speech recognition (ASR). Figure 28.1 gives simple, familiar examples of weighted automata as used in ASR. The automaton in Fig. 28.1a is a toy finite-state *language model*. The legal word strings are specified by the words along each complete path, and their probabilities by the product of the corresponding transition probabilities. The automaton in Fig. 28.1b gives the possible pronunciations of one word, 'data', used in the language model. Each legal pronunciation is the phone string along a complete path, and its probability is given by the product of the corresponding transition probabilities. Finally, the automaton in Fig. 28.1c encodes a typical left-to-right, three-distribution HMM structure for one phone, with the labels along a complete path specifying legal strings of acoustic distributions for that phone.

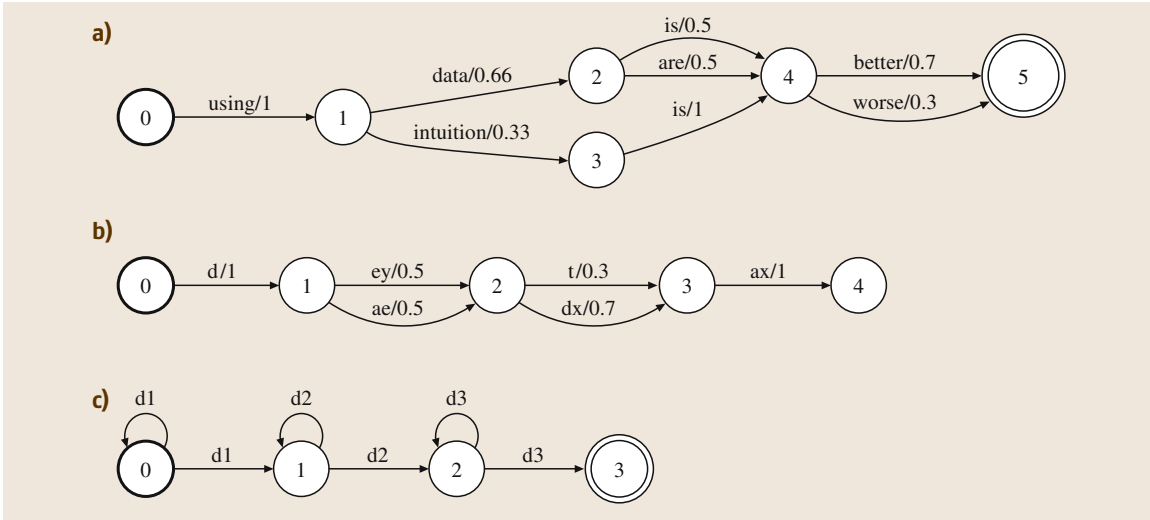


Fig. 28.1a–c Weighted finite-state acceptor examples. By convention, the states are represented by *circles* and marked with their unique number. The initial state is represented by a *bold circle*, final states by *double circles*. The label l and weight w of a transition are marked on the corresponding directed arc by l/w. When explicitly shown, the final weight w of a final state f is marked by f/w

These automata consist of a set of states, an initial state, a set of final states (with final weights), and a set of transitions between states. Each transition has a source state, a destination state, a label, and a weight. Such automata are called *weighted finite-state acceptors* (WFSAs), since they *accept* or *recognize* each string that can be read along a path from the start state to a final state. Each accepted string is assigned a weight, namely the accumulated weights along accepting paths for that string, including the final weights. An acceptor as a whole represents a set of strings, namely those that it accepts. As a weighted acceptor, it also associates to each accepted string the accumulated weights of their accepting paths.

Speech recognition architectures commonly give the run-time decoder the task of combining and optimizing automata such as those in Fig. 28.1. The decoder finds word pronunciations in its lexicon and substitutes them into the grammar. Phonetic tree representations may be used at this point to reduce path redundancy and thus improve search efficiency, especially for large-vocabulary recognition [28.11]. The decoder then identifies the correct context-dependent models to use for each phone in context, and finally substitutes them to create an HMM-level transducer. The software that performs these operations is usually tied to particular model topologies. For example, the context-dependent models might have to be triphonic, the grammar might be restricted to trigrams, and the alternative pronunciations might have to be enumerated in the lexicon. In addition, these automata combinations and optimizations are applied in a preprogrammed order to a prespecified number of levels.

28.2.2 Weighted Transducers

Our approach uses finite-state transducers, rather than acceptors, to represent the n-gram grammars, pronunciation dictionaries, context-dependency specifications, HMM topology, word, phone or HMM segmentations, lattices, and n-best output lists encountered in ASR. The transducer representation provides general methods for combining models and optimizing them, leading to both simple and flexible ASR decoder designs.

A weighted finite-state transducer (WFST) is quite similar to a weighted acceptor except that it has an input label, an output label, and a weight on each of its transitions. The examples in Fig. 28.2 encode (a superset of) the information in the WFSAs of Fig. 28.1a,b as WFSTs. Figure 28.2a represents the same language model as Fig. 28.1a by giving each transition identical input and output labels. This adds no new information, but is a convenient way we often use to treat acceptors and transducers uniformly.

Figure 28.2b represents a toy pronunciation lexicon as a mapping from phone strings to words in the lexicon, in this example 'data' and 'dew', with probabilities representing the likelihoods of alternative pronunciations. It *transduces* a phone string that can be read along a path from the start state to a final state to a word string with a particular weight. The word corresponding to a pronunciation is output by the transition that consumes the

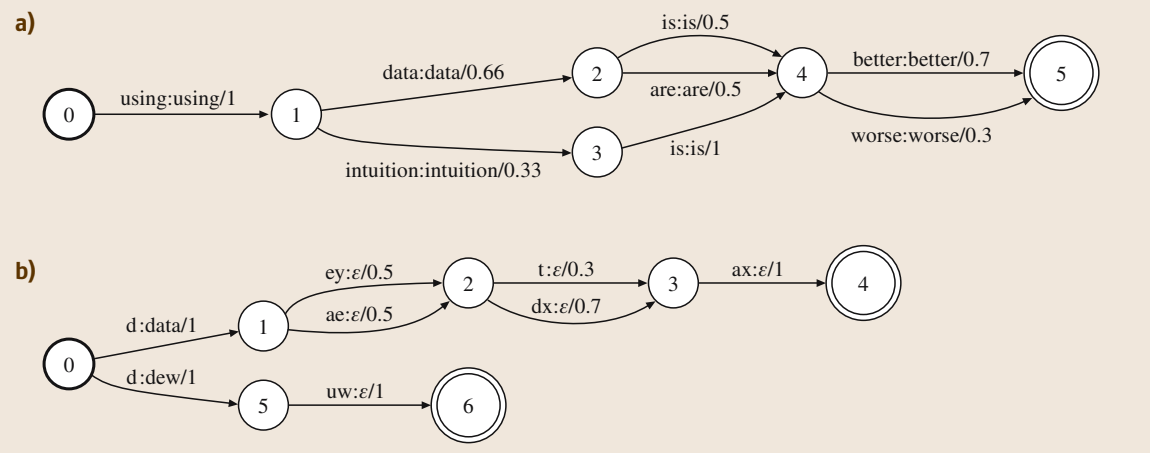


Fig. 28.2a,b Weighted finite-state transducer examples. These are similar to the weighted acceptors in Fig. 28.1 except output labels are introduced in each transition. The input label i, the output label o, and weight w of a transition are marked on the corresponding directed arc by $i : o/w$

first phone for that pronunciation. The transitions that consume the remaining phones output no further symbols, indicated by the null symbol ε as the transition's output label. In general, an ε input label marks a transition that consumes no input, and an ε output label marks a transition that produces no output.

This transducer has more information than the WFSA in Fig. 28.1b. Since words are encoded by the output label, it is possible to combine the pronunciation transducers for more than one word without losing word identity. Similarly, HMM structures of the form given in Fig. 28.1c can be combined into a single transducer that preserves phone model identity. This illustrates the key advantage of a transducer over an acceptor: the transducer can represent a relationship between two levels of representation, for instance between phones and words or between HMMs and context-independent phones. More precisely, a transducer specifies a binary relation between strings: two strings are in the relation when there is a path from an initial to a final state in the transducer that has the first string as the sequence of input labels along the path and the second string as the sequence of output labels along the path (ε symbols are left out in both input and output). In general, this is a relation rather than a function since the same input string might be transduced to different strings along two distinct paths. For a weighted transducer, each string pair is also associated with a weight.

We rely on a common set of weighted transducer operations to combine, optimize, search, and prune them [28.4]. Each operation implements a single, well-defined function that has its foundations in the mathematical theory of rational power series [28.12–14]. Many of those operations are the weighted transducer generalizations of classical algorithms for unweighted acceptors. We have brought together those and a variety of auxiliary operations in a comprehensive weighted finite-state machine software library (FsmLib) available for non-commercial use from the AT&T Labs – research web site [28.4]. The OpenFST open-source library in C++ provides a new implementation of the main weighted transducer algorithms [28.15].

Basic *union*, *concatenation*, and *Kleene closure* operations combine transducers in parallel, in series, and with arbitrary repetition, respectively. Other operations convert transducers to acceptors by projecting onto the input or output label set, find the best or the n best paths in a weighted transducer, remove unreachable states and transitions, and sort acyclic automata topologically.

Where possible, we provided *lazy* (also called *on-demand*) implementations of algorithms. Any finite-state object `fsm` can be accessed with the three main methods `fsm.start()`, `fsm.final(state)`, and `fsm.transitions(state)` that return the start state, the final weight of a state, and the transitions leaving a state, respectively. This interface can be implemented for concrete automata in an obvious way: the methods simply return the requested information from a stored representation of the automaton. However, the interface can also be given lazy implementations. For example, the lazy union of two automata returns a new lazy `fsm` object. When the object is first constructed, the lazy implementation just initializes internal bookkeeping data. It is only when the interface methods request the start state, the final weights, or the transitions (and their destination states) leaving a state, that this information is actually computed, and optionally cached inside the object for later reuse. This approach has the advantage that, if only a part of the result of an operation is needed (for example in a pruned search), then the unused part is never computed, saving time and space. We refer the interested reader to the library documentation and an overview of the library [28.4] for further details on lazy finite-state objects.

We now discuss the key transducer operations that are used in our speech applications for model combination, redundant-path removal, and size reduction.

28.2.3 Composition

Composition is the transducer operation for combining different levels of representation. For instance, a pronunciation lexicon can be composed with a word-level grammar to produce a phone-to-word transducer whose word strings are restricted to the grammar. A variety of ASR transducer combination techniques, both context independent and context dependent, can be conveniently and efficiently implemented with composition.

As previously noted, a transducer represents a binary relation between strings. The composition of two transducers represents their relational composition. In particular, the composition $T = T_1 \circ T_2$ of two transducers T_1 and T_2 has exactly one path mapping string u to string w for each pair of paths, the first in T_1 mapping u to some string v and the second in T_2 mapping v to w. The weight of a path in T is computed from the weights of the two corresponding paths in T_1 and T_2 with the same operation that computes the weight of a path from the weights of its transitions. If the transition weights represent probabilities, that operation is the product. If instead the weights represent log probabilities or negative log probabilities as is common in

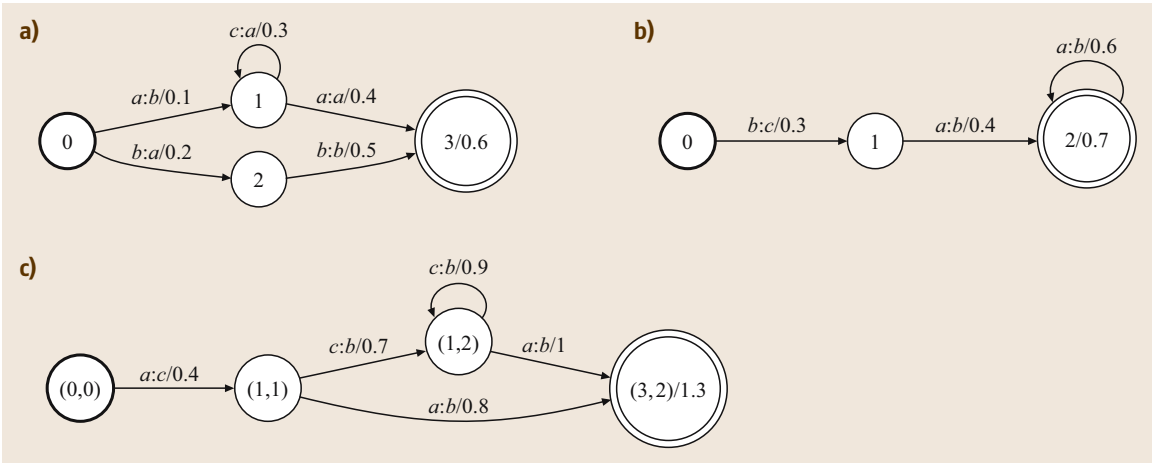


Fig. 28.3a–c Example of transducer composition

ASR for numerical stability, the operation is the sum. More generally, the weight operations for a weighted transducer can be specified by a *semiring* [28.12–14], as discussed in more detail in Sect. 28.3.

The weighted composition algorithm generalizes the classical state-pair construction for finite automata intersection [28.16] to weighted acceptors and transducers. The states of the composition T are pairs of a T_1 state and a T_2 state. T satisfies the following conditions:

1. its initial state is the pair of the initial state of T_1 and the initial state of T_2;
2. its final states are pairs of a final state of T_1 and a final state of T_2, and
3. there is a transition t from (q_1, q_2) to (r_1, r_2) for each pair of transitions t_1 from q_1 to r_1 and t_2 from q_2 to r_2 such that the output label of t_1 matches the input label of t_2.

The transition t takes its input label from t_1, its output label from t_2, and its weight is the combination of the weights of t_1 and t_2 done with the same operation that combines weights along a path. Since this computation is *local* – it involves only the transitions leaving two states being paired – it can be given a lazy implementation in which the composition is generated only as needed by other operations on the composed automaton. Transitions with ε-labels in T_1 or T_2 must be treated specially, as discussed in Sect. 28.3. Figure 28.3a and b show two simple transducers and the result of their composition, Fig. 28.3c. The weight of a path in the resulting transducer is the sum of the weights of the matching paths in T_1 and T_2 (as when the weights represent negative log probabilities).

Since we represent weighted acceptors by weighted transducers in which the input and output labels of each transition are identical, the intersection of two weighted acceptors is just the composition of the corresponding transducers.

28.2.4 Determinization

In a *deterministic automaton*, each state has at most one transition with any given input label and there are no input ε-labels. Figure 28.4a gives an example of a nondeterministic weighted acceptor: at state 0, for instance, there are two transitions with the same label a. The automaton in Fig. 28.4b, on the other hand, is deterministic.

The key advantage of a deterministic automaton over equivalent nondeterministic ones is its irredundancy: it contains at most one path matching any given input string, thereby reducing the time and space needed to process the string. This is particularly important in ASR due to pronunciation lexicon redundancy in large-vocabulary tasks. The familiar tree lexicon in ASR is a deterministic pronunciation representation [28.11].

To benefit from determinism, we use a determinization algorithm that transforms a nondeterministic weighted automaton into an equivalent deterministic automaton. Two weighted acceptors are *equivalent* if they associate the same weight to each input string; weights may be distributed differently along the paths of two equivalent acceptors. Two weighted transducers are equivalent if they associate the same output string and weights to each input string; the distribution of the

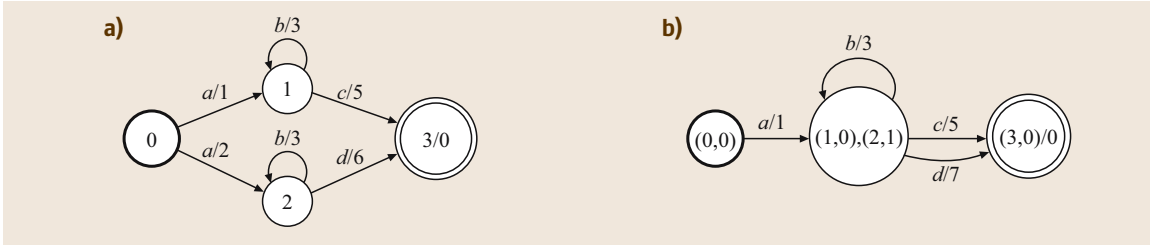


Fig. 28.4a,b Determinization of weighted automata. **(a)** Weighted automaton over the tropical semiring A. **(b)** Equivalent weighted automaton B obtained by the determinization of A

weight or output labels along paths need not be the same in the two transducers.

If we apply the weighted determinization algorithm to the union of a set of chain automata, each representing a single word pronunciation, we obtain a tree-shaped automaton. However, the result of this algorithm on more-general automata may not be a tree, and in fact may be much more compact than a tree. The algorithm can produce results for a broad class of automata with cycles that have no tree representation.

Weighted determinization generalizes the classical subset method for determinizing unweighted finite automata [28.17]. Unlike in the unweighted case, not all weighted automata can be determinized. Conditions for determinizability are discussed in Sect. 28.3.3. Fortunately, most weighted automata used in speech processing can be either determinized directly or easily made determinizable with simple transformations, as we discuss in Sects. 28.3.3 and 28.4.1. In particular, any acyclic weighted automaton can be determinized.

To eliminate redundant paths, weighted determinization needs to calculate the combined weight of all paths with the same labeling. When each path represents a disjoint event with a probability given by its weight, the combined weight, representing the probability of the common labeling for that set of paths, would be the sum of the weights of the paths. Alternatively, we may just want to keep the most probable path, as is done in shortest path algorithms, leading to the so-called *Viterbi approximation*. When weights are negative log probabilities, these two alternatives correspond respectively to log summation and taking the minimum. In the general case, we use one operation, denoted $\otimes$, for combining weights along paths and for composition, and a second operation, denoted $\oplus$, to combine identically labeled paths. Some common choices of $(\oplus, \otimes)$ include $(\max, +)$, $(+, *)$, $(\min, +)$, and $(-\log(e^{-x} + e^{-y}), +)$. In speech applications, the first two are appropriate for probabilities, the last two for the corresponding negative log probabilities. More generally, as we will see in Sect. 28.3, many of the weighted automata algorithms apply when the two operations define an appropriate semiring. The choices $(\min, +)$, and $(-\log(e^{-x} + e^{-y}), +)$ are called the *tropical* and log semirings, respectively.

Our discussion and examples of determinization and, later, minimization will be illustrated with weighted acceptors. The *string* semiring, whose two operations are longest common prefix and concatenation, can be used to treat the output strings as weights. By this method, the transducer case can be handled as well; see *Mohri* [28.6] for details.

We will now work through an example of determinization with weights in the tropical semiring. Figure 28.4b shows the weighted determinization of automaton A_1 from Fig. 28.4a. In general, the determinization of a weighted automaton is equivalent to the original, that is, it associates the same weight to each input string. For example, there are two paths corresponding to the input string ab in A_1, with weights $\{1+3=4, 2+3=5\}$. The minimum 4 is also the weight associated by A_2 to the string ab.

In the classical subset construction for determinizing unweighted automata, all the states reachable by a given input from the initial state are placed in the same subset. In the weighted case, transitions with the same input label can have different weights, but only the minimum of those weights is needed and the leftover weights must be tracked. Thus, the subsets in weighted determinization contain pairs (q, w) of a state q of the original automaton and a leftover weight w.

The initial subset is $[(i, 0)]$, where i is the initial state of the original automaton. For example, in Fig. 28.4, the initial subset for automaton B is $[(0, 0)]$. Each new subset S is processed in turn. For each element a of the input alphabet Σ labeling at least one transition leaving a state of S, a new transition t leaving S is constructed

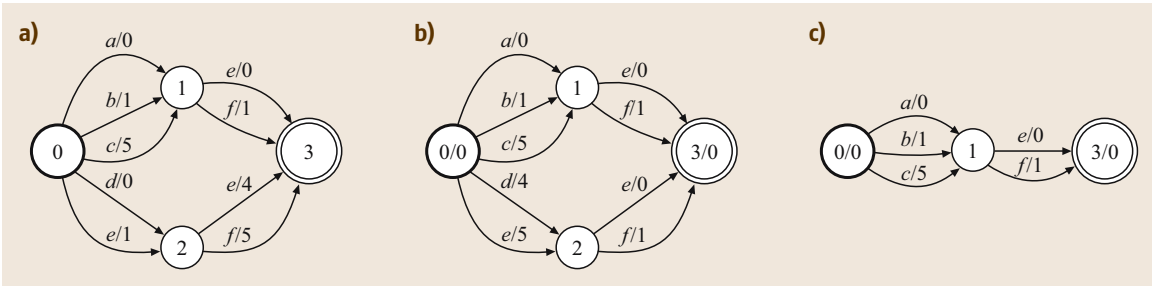


Fig. 28.5a–c Weight pushing and minimization. **(a)** Deterministic weighted automaton A. **(b)** Equivalent weighted automaton B obtained by weight pushing in the tropical semiring. **(c)** Minimal weighted automaton equivalent to A

in the result automaton. The input label of t is a and its weight is the minimum of the sums $w+l$, where w is s's leftover weight, and l is the weight of an a-transition leaving a state s in S. The destination state of t is the subset S' containing those pairs (q', w') in which q' is a state reached by a transition labeled with a from a state of S and w' is the appropriate leftover weight.

For example, in Fig. 28.4, the transition leaving $(0, 0)$ in B labeled with a is obtained from the two transitions labeled with a leaving state 0 in A: its weight is the minimum of the weight of those two transitions, and its destination state is the subset $S' = \{(1, 1-1=0), (2, 2-1=1)\}$. The algorithm is described in more in detail in Sect. 28.3.3.

It is clear that the transitions leaving a given state in the determinization of an automaton can be computed from the subset for the state and the transitions leaving the states in the subset, as is the case for the classical nondeterministic finite automata (NFA) determinization algorithm. In other words, the weighted determinization algorithm is locally like the composition algorithm, and can thus be given a lazy implementation that creates states and transitions only as needed.

28.2.5 Minimization

Given a deterministic automaton, we can reduce its size by minimization, which can save both space and time. Any deterministic unweighted automaton can be minimized using classical algorithms [28.18, 19]. In the same way, any deterministic weighted automaton A can be minimized using our minimization algorithm, which extends the classical algorithm [28.6]. The resulting weighted automaton B is equivalent to the automaton A, and has the least number of states and the least number of transitions among all deterministic weighted automata equivalent to A.

As we will see in Sect. 28.3.5, weighted minimization is quite efficient, indeed as efficient as classical deterministic finite automata (DFA) minimization.

We can view the deterministic weighted automaton A in Fig. 28.5a as an unweighted automaton by interpreting each pair (a, w) of a label a and a weight w as a single label. We can then apply the standard DFA minimization algorithm to this automaton. However, since the pairs for different transitions are all distinct, classical minimization would have no effect on A.

The size of A can still be reduced by using true weighted minimization. This algorithm works in two steps: the first step *pushes* weight among transitions, and the second applies the classical minimization algorithm to the result with each distinct label–weight pair viewed as a distinct symbol, as described above. Weight pushing is useful not only as a first step of minimization but also to redistribute weight among transitions to improve search, especially pruned search. The algorithm is described in more detail in Sect. 28.3.4, and analyzed in [28.20]. Its applications to speech recognition are discussed in [28.21].

Pushing is a special case of *reweighting*. We describe reweighting in the case of the tropical semiring; similar definitions can be given for other semirings. A (nontrivial) weighted automaton can be reweighted in an infinite number of ways that produce equivalent automata. To see how, let i be A's initial state and assume for convenience that A has a single final state f. Any automaton can be transformed into an equivalent automaton with a single final state by adding a super-final state, making all previously final states non-final, and adding from each previously final state f with weight $\rho(f)$ an ε-transition with the weight $\rho(f)$ to the super-final state. Let $V : Q \to \mathbb{R}$ be an arbitrary *potential* function on states. Update each weight $w[t]$ of the transition t from state $p[t]$ to $n[t]$ as

follows:

$$w[t] \leftarrow w[t] + [V(n[t]) - V(p[t])] , \tag{28.1}$$

and the final weight $\rho(f)$ as

$$\rho(f) \leftarrow \rho(f) + [V(i) - V(f)] . \tag{28.2}$$

It is easy to see that with this reweighting each potential internal to any successful path from the initial state to the final state is added and then subtracted, making the overall change in path weight:

$$[V(f) - V(i)] + [V(i) - V(f)] = 0 . \tag{28.3}$$

Thus, reweighting does not affect the total weight of a successful path and the resulting automaton is equivalent to the original.

To push the weight in A towards the initial state as much as possible, a specific potential function is chosen, the one that assigns to each state the lowest path weight from that state to the final state. After pushing, the lowest-cost path (excluding the final weight) from every state to the final state will thus be 0.

Figure 28.5b shows the result of pushing for the input A. Thanks to pushing, the size of the automaton can then be reduced using classical minimization. Figure 28.5c illustrates the result of the final step of the algorithm. No approximation or heuristic is used: the resulting automaton C is equivalent to A.

28.2.6 Speech Recognition Transducers

As an illustration of these methods applied to speech recognition, we describe how to construct a single, statically compiled and optimized recognition transducer that maps from context-dependent phones to words. This is an attractive choice for tasks that have fixed acoustic, lexical, and grammatical models since the static transducer can be searched simply and efficiently with no recognition-time overhead for model combination and optimization.

Consider the pronunciation lexicon in Fig. 28.2b. Suppose we form the union of this transducer with the pronunciation transducers for the remaining words in the grammar G of Fig. 28.2a by creating a new super-initial state and connecting an ε-transition from that state to the former start states of each pronunciation transducer. We then take its Kleene closure by connecting an ε-transition from each final state to the initial state. The resulting pronunciation lexicon L would pair any word string from that vocabulary to their corresponding pronunciations. Thus,

$$L \circ G \tag{28.4}$$

gives a transducer that maps from phones to word strings restricted to G.

We used composition here to implement a context-independent substitution. However, a major advantage of transducers in speech recognition is that they naturally generalize the notion of context-independent substitution of a label to the context-dependent case. In particular, the application of the familiar triphone models in ASR to the context-independent transducer, producing a context-dependent transducer, can be performed simply with composition.

To do so, we first construct a context-dependency transducer that maps from context-independent phones to context-dependent triphones. This transducer has a state for every pair of phones and a transition for every context-dependent model. In particular, if ae/k_t represents the triphonic model for ae with left context k and right context t, then there is a transition in the context-dependency transducer from state (k, ae) to state (ae, t) with output label ae/k_t. This use of / to indicate *in the context of* in a triphone symbol offers a potential ambiguity with our use of / to separate a transition's weight from its input and output symbols. However, since context-dependency transducers are never weighted in this chapter, the confusion is not a problem in what follows, so we chose to stay with the notation of previous work rather than changing it to avoid the potential ambiguity. For the input label on this transition, we could choose the center phone ae as depicted in Fig. 28.6a. This will correctly implement the transduction; but the transducer will be non-deterministic. Alternately, we can choose the right phone t as depicted in Fig. 28.6b. This will also correctly implement the transduction, but the result will be deterministic. To see why these are correct, realize that when we enter a state, we have read (in the deterministic case) or must read (in the nondeterministic case) the two phones that label the state. Therefore, the source state and destination state of a transition determine the triphone context. In Sect. 28.4, we give the full details of the triphonic

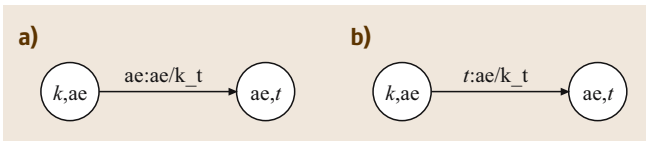


Fig. 28.6a,b Context-dependent triphone transducer transition: **(a)** nondeterministic, **(b)** deterministic

context-dependency transducer construction and further demonstrate its correctness.

The above context-dependency transducer maps from context-independent phones to context-dependent triphones. We can invert the relation by interchanging the transducer's input and output labels to create a new transducer that maps from context-dependent triphones to context-independent phones. We do this inversion so we can left compose it with our context-independent recognition transducer $L \circ G$. If we let C represent a context-dependency transducer from context-dependent phones to context-independent phones, then:

$$C \circ (L \circ G) \tag{28.5}$$

gives a transducer that maps from context-dependent phones to word strings restricted to the grammar G. To complete our example, we optimize this transducer. Given our discussion of the benefits of determinization and minimization, we might try to apply those operations directly to the composed transducer:

$$N = \min(\det(C \circ (L \circ G))) \,. \tag{28.6}$$

This assumes that the recognition transducer can be determinized, which will be true if each of the component transducers can be determinized. If the context dependency C is constructed as we have described and if the grammar G is an n-gram language model, then they will be determinizable. However, L may not be determinizable. In particular, if L has ambiguities, namely homophones (two distinct words that have the same pronunciation), then it cannot be determinized as is. However, we can introduce auxiliary phone symbols at word ends to disambiguate homophones to create a transformed lexicon $\tilde{L}$. We also need to create a modified context-dependency transducer $\tilde{C}$ that additionally pairs the context-independent auxiliary symbols found in the lexicon with new context-dependent auxiliary symbols (which are later rewritten as epsilons after all optimizations). We leave the details to Sect. 28.4. The following expression specifies the optimized transducer:

$$N = \min(\det(\tilde{C} \circ (\tilde{L} \circ G))) \,. \tag{28.7}$$

In Sect. 28.4, we give illustrative experimental results with a fully composed, optimized (and *factored*) recognition transducer that maps from context-dependent units to words for the North American Business News (NAB) DARPA task. This transducer runs about 18 times faster than its unoptimized version and has only about 1.4× times as many transitions as its word-level grammar. We have found similar results with DARPA Broadcast News and Switchboard.

28.3 Algorithms

We now describe in detail the weighted automata and transducer algorithms introduced informally in Sect. 28.2 that are relevant to the design of speech recognition systems. We start with definitions and notation used in specifying and describing the algorithms.

28.3.1 Preliminaries

As noted earlier, all of our algorithms work with weights that are combined with operations satisfying the *semiring* conditions. A semiring $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ is specified by a set of values $\mathbb{K}$, two binary operations $\oplus$ and $\otimes$, and two designated values $\overline{0}$ and $\overline{1}$. The operation $\oplus$ is associative, commutative, and has $\overline{0}$ as identity. The operation $\otimes$ is associative, has identity $\overline{1}$, distributes with respect to $\oplus$, and has $\overline{0}$ as annihilator: for all $a \in \mathbb{K}$, $a \otimes \overline{0} = \overline{0} \otimes a = \overline{0}$. If $\otimes$ is also commutative, we say that the semiring is *commutative*. All the semirings we discuss in the rest of this chapter are commutative.

Real numbers with addition and multiplication satisfy the semiring conditions, but of course they also satisfy several other important conditions (for exam-

Table 28.1 Semiring examples. $\oplus_{\log}$ is defined by: $x \oplus_{\log} y = -\log(\mathrm{e}^{-x} + \mathrm{e}^{-y})$

Semiring	Set	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Boolean	$\{0, 1\}$	$\vee$	$\wedge$	0	1
Probability	$\mathbb{R}_+$	$+$	$\times$	0	1
Log	$\mathbb{R} \cup \{-\infty, +\infty\}$	$\oplus_{\log}$	$+$	$+\infty$	0
Tropical	$\mathbb{R} \cup \{-\infty, +\infty\}$	min	$+$	$+\infty$	0

ple, having additive inverses), which are not required for our transducer algorithms. Table 28.1 lists some familiar (commutative) semirings. In addition to the Boolean semiring, and the probability semiring used to combine probabilities, two semirings often used in text and speech processing applications are the *log semiring*, which is isomorphic to the probability semiring via the negative-log mapping, and the *tropical semiring*, which is derived from the log semiring using the *Viterbi approximation*.

A semiring $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ is *weakly left-divisible* if for any x and y in $\mathbb{K}$ such that $x \oplus y \neq \overline{0}$, there exists at least one z such that $x = (x \oplus y) \otimes z$. The $\otimes$-operation is *cancellative* if z is unique and we can write: $z = (x \oplus y)^{-1} \otimes x$. A semiring is *zero-sum-free* if, for any x and y in $\mathbb{K}$, $x \oplus y = \overline{0}$ implies $x = y = \overline{0}$.

For example, the tropical semiring is weakly left-divisible with $z = x - \min(x, y)$, which also shows that $\otimes$ for this semiring is cancellative. The probability semiring is also weakly left-divisible with $z = \frac{x}{x+y}$. Finally, the tropical semiring, the probability semiring, and the log semiring are zero sum free.

For any $x \in \mathbb{K}$, let x^n denote

$$x^n = \underbrace{x \otimes \cdots \otimes x}_{n} \,. \tag{28.8}$$

When the infinite sum $\bigoplus_{n=0}^{+\infty} x^n$ is well defined and in $\mathbb{K}$, the closure of an element $x \in \mathbb{K}$ is defined as $x^* = \bigoplus_{n=0}^{+\infty} x^n$. A semiring is *closed* when infinite sums such as the one above, are well defined and if associativity, commutativity, and distributivity apply to countable sums (*Lehmann* [28.20, 22] and *Mohri* give precise definitions). The Boolean and tropical semirings are closed, while the probability and log semirings are not.

A *weighted finite-state transducer* $T = (\mathcal{A}, \mathcal{B}, Q, I, F, E, \lambda, \rho)$ over a semiring $\mathbb{K}$ is specified by a finite input alphabet $\mathcal{A}$, a finite output alphabet $\mathcal{B}$, a finite set of states Q, a set of initial states $I \subseteq Q$, a set of final states $F \subseteq Q$, a finite set of transitions $E \subseteq Q \times (\mathcal{A} \cup \{\varepsilon\}) \times (\mathcal{B} \cup \{\varepsilon\}) \times \mathbb{K} \times Q$, an initial state weight assignment $\lambda : I \to \mathbb{K}$, and a final state weight assignment $\rho : F \to \mathbb{K}$. $E[q]$ denotes the set of transitions leaving state $q \in Q$. $|T|$ denotes the sum of the number of states and transitions of T.

Weighted automata (or weighted acceptors) are defined in a similar way by simply omitting the input or output labels. The *projection* operations $\Pi_1(T)$ and $\Pi_2(T)$ obtain a weighted automaton from a weighted transducer T by omitting, respectively, the input or the output labels of T.

Given a transition $e \in E$, $p[e]$ denotes its origin or previous state, $n[e]$ its destination or next state, $i[e]$ its input label, $o[e]$ its output label, and $w[e]$ its weight. A *path* $\pi = e_1 \cdots e_k$ is a sequence of consecutive transitions: $n[e_{i-1}] = p[e_i], i = 2, \ldots, k$. The path π is a *cycle* if $p[e_1] = n[e_k]$. An *ε-cycle* is a cycle in which the input and output labels of all transitions are ε.

The functions n, p, and w on transitions can be extended to paths by setting $n[\pi] = n[e_k]$ and $p[\pi] = p[e_1]$, and by defining the weight of a path as the $\otimes$-product of the weights of its constituent transitions: $w[\pi] = w[e_1] \otimes \cdots \otimes w[e_k]$. More generally, w is extended to any finite set of paths R by setting $w[R] = \bigoplus_{\pi \in R} w[\pi]$; if the semiring is closed, this is defined even for infinite R. We denote by $P(q, q')$ the set of paths from q to q' and by $P(q, x, y, q')$ the set of paths from q to q' with input label $x \in \mathcal{A}^*$ and output label $y \in \mathcal{B}^*$. For an acceptor, we denote by $P(q, x, q')$ the set of paths with input label x. These definitions can be extended to subsets $R, R' \subseteq Q$ by $P(R, R') = \cup_{q \in R,\, q' \in R'} P(q, q')$, $P(R, x, y, R') = \cup_{q \in R,\, q' \in R'} P(q, x, y, q')$, and, for an acceptor, $P(R, x, R') = \cup_{q \in R,\, q' \in R'} P(q, x, q')$. A transducer T is *regulated* if the weight associated by T to any pair of input–output strings (x, y), given by

$$T(x, y) = \bigoplus_{\pi \in P(I, x, y, F)} \lambda[p[\pi]] \otimes w[\pi] \otimes \rho[n[\pi]] \,, \tag{28.9}$$

is well defined and in $\mathbb{K}$. If $P(I, x, y, F) = \emptyset$, then $T(x, y) = \overline{0}$. A weighted transducer without ε-cycles is regulated, as is any weighted transducer, over a closed semiring. Similarly, for a regulated acceptor, we define

$$T(x) = \bigoplus_{\pi \in P(I, x, F)} \lambda[p[\pi]] \otimes w[\pi] \otimes \rho[n[\pi]] \,. \tag{28.10}$$

The transducer T is *trim* if every state occurs in some path $\pi \in P(I, F)$. In other words, a trim transducer has no useless states. The same definition applies to acceptors.

28.3.2 Composition

As we outlined in Sect. 28.2.3, composition is the core operation for relating multiple levels of representation in ASR. More generally, composition is the fundamental algorithm used to create complex weighted transducers from simpler ones [28.12, 14], and generalizes the composition algorithm for unweighted finite-state transducers [28.23, 24]. Let $\mathbb{K}$ be a commutative semiring and

let T_1 and T_2 be two weighted transducers defined over $\mathbb{K}$ such that the input alphabet $\mathcal{B}$ of T_2 coincides with the output alphabet of T_1. Assume that the infinite sum $\bigoplus_{z\in\mathcal{B}^*} T_1(x,z)\otimes T_2(z,y)$ is well defined and in $\mathbb{K}$ for all $(x,y)\in\mathcal{A}^*\times\mathcal{C}^*$, where $\mathcal{A}$ is the input alphabet of T_1 and $\mathcal{C}$ is the output alphabet of T_2. This will be the case if $\mathbb{K}$ is closed, or if T_1 has no ε-input cycles or T_2 has no ε-output cycles. Then, the result of the composition of T_1 and T_2 is a weighted transducer denoted by $T_1\circ T_2$ and specified for all x, y by:

$$(T_1\circ T_2)(x,y)=\bigoplus_{z\in\mathcal{B}^*} T_1(x,z)\otimes T_2(z,y)\,. \qquad (28.11)$$

There is a general and efficient composition algorithm for weighted transducers [28.12, 14]. States in the composition $T_1\circ T_2$ of two weighted transducers T_1 and T_2 are identified with pairs of a state of T_1 and a state of T_2. Leaving aside transitions with ε inputs or outputs, the following rule specifies how to compute a transition of $T_1\circ T_2$ from appropriate transitions of T_1 and T_2:

$$(q_1,a,b,w_1,r_1)\text{ and }(q_2,b,c,w_2,r_2)\Longrightarrow((q_1,q_2),a,c,w_1\otimes w_2,(r_1,r_2))\,. \qquad (28.12)$$

Figure 28.7 gives the pseudocode of the algorithm in the ε-free case.

The algorithm takes as input two weighted transducers

$$T_1=(\mathcal{A},\mathcal{B},Q_1,I_1,F_1,E_1,\lambda_1,\rho_1)\quad\text{and}$$
$$T_2=(\mathcal{B},\mathcal{C},Q_2,I_2,F_2,E_2,\lambda_2,\rho_2)\,, \qquad (28.13)$$

and outputs a weighted finite-state transducer $T=(\mathcal{A},\mathcal{C},Q,I,F,E,\lambda,\rho)$ implementing the composition of T_1 and T_2. E, I, and F are all initialized to the empty set and grown as needed.

The algorithm uses a queue S containing the set of pairs of states yet to be examined. The queue discipline of S is arbitrary, and does not affect the termination of the algorithm. The state set Q is initially the set of pairs of initial states of the original transducers, as is S (lines 1–2). Each time through the loop in lines 3–16, a new pair of states (q_1,q_2) is extracted from S (lines 4–5). The initial weight of (q_1,q_2) is computed by $\otimes$-multiplying the initial weights of q_1 and q_2 when they are both initial states (lines 6–8). Similar steps are followed for final states (lines 9–11). Then, for each pair of matching transitions (e_1,e_2), a new transition is created according to the rule specified earlier (line 16). If the destination state $(n[e_1],n[e_2])$ has not been found previously, it is added to Q and inserted in S (lines 14–15).

```
Weighted-composition (T1, T2)
 1  Q ← I1 × I2
 2  S ← I1 × I2
 3  while S ≠ ∅ do
 4       (q1, q2) ← Head (S)
 5       Dequeue (S)
 6       if (q1, q2) ∈ I1 × I2 then
 7            I ← I ∪ {(q1, q2)}
 8            λ (q1, q2) ← λ1 (q1) ⊗ λ2 (q2)
 9       if (q1, q2) ∈ F1 × F2 then
10            F ← F ∪ {(q1, q2)}
11            ρ (q1, q2) ← ρ1 (q1) ⊗ ρ2 (q2)
12       for each (e1, e2) ∈ E [q1] × E [q2] such that o [e1] = i [e2] do
13            if (n [e1], n [e2]) ∉ Q then
14                 Q ← Q ∪ {(n[e1], n[e2])}
15                 Enqueue (S, (n[e1], n[e2]))
16            E ← E ∪ {((q1, q2,), i[e1], o[e2], w[e1] ⊗ w[e2], (n[e1], n[e2]))}
17  return T
```

Fig. 28.7 Pseudocode for the composition algorithm

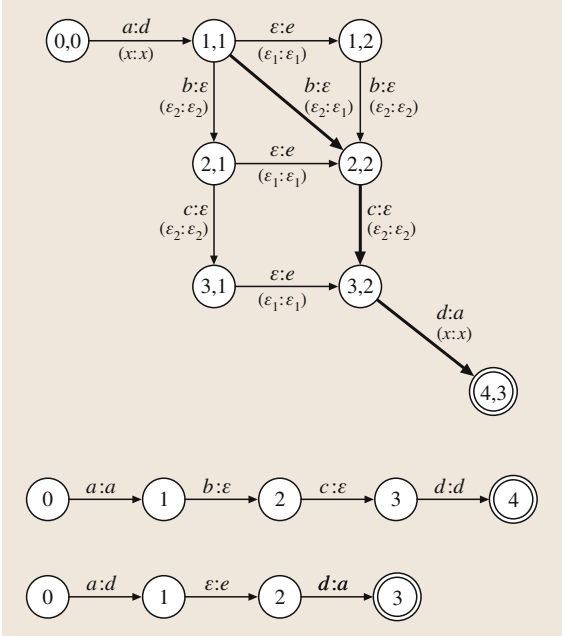


Fig. 28.8 Redundant ε paths. A straightforward generalization of the ε-free case could generate all the paths from (1, 1) to (3, 2) when composing the two simple transducers on the right-hand side

In the worst case, all transitions of T_1 leaving a state q_1 match all those of T_2 leaving state q_1', thus the space and time complexity of composition is quadratic: $O(|T_1||T_2|)$. However, a lazy implementation of composition can be used to construct just the part of the composed transducer that is needed.

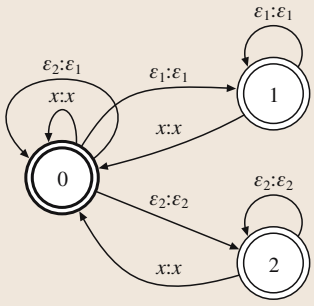


Fig. 28.9 Filter for composition F

More care is needed when T_1 has output ε labels and T_2 input ε labels. Indeed, as illustrated by Fig. 28.8, a straightforward generalization of the ε-free case would generate redundant ε-paths and, in the case of non-idempotent semirings, would lead to an incorrect result. The weight of the matching paths of the original transducers would be counted p times, where p is the number of redundant paths in the composition.

To solve this problem, all but one ε-path must be filtered out of the composition. Figure 28.8 indicates in thick lines one possible choice for that path, which in this case is the shortest. Remarkably, that filtering mechanism can be encoded as a finite-state transducer.

Let $\tilde{T}_1$ and $\tilde{T}_2$ be the weighted transducers obtained from T_1 and T_2, respectively, by replacing the output ε labels of T_1 with ε_2 and the input ε labels of T_2 with ε_1. Consider the filter finite-state transducer F represented in Fig. 28.9. Then $\tilde{T}_1 \circ F \circ \tilde{T}_2 = T_1 \circ T_2$. Since the two compositions in $\tilde{T}_1 \circ F \circ \tilde{T}_2$ do not involve ε labels, the ε-free composition already described can be used to compute the resulting transducer.

Intersection (or the *Hadamard product*) of weighted automata and composition of finite-state transducers are both special cases of composition of weighted transducers. Intersection corresponds to the case where the input and output labels of transitions are identical and composition of unweighted transducers is obtained simply by omitting the weights.

```
Weighted-determinization (A)
 1  i' ← {(i, λ(i)) : i ∈ I}
 2  λ'(i') ← 1̄
 3  S ← {i'}
 4  while S ≠ ∅ do
 5      p' ← Head (S)
 6      Dequeue (S)
 7      for each x ∈ i[E[Q[p']]] do
 8          w' ← ⊕ {v ⊗ w : (p, v) ∈ p', (p, x, w, q) ∈ E}
 9          q' ← {(q, ⊕ {(w'^{-1} ⊗ (v ⊗ w) : (p, v) ∈ p', (p, x, w, q) ∈ E}) :
                q = n[e], i[e] = x, e ∈ E[Q[p']]}
10          E' ← E' ∪ {(p', x, w', q')}
11          if q' ∉ Q then
12              Q ← Q' ∪ {q'}
13              if Q[q'] ∩ F ≠ ∅ then
14                  F' ← F' ∪ {q'}
15                  ρ'(q') ← ⊕ {v ⊗ ρ(q) : (q, v) ∈ q', q ∈ F}
16              Enqueue (S, q')
17  return A'
```

Fig. 28.10 Pseudocode for the weighted determinization algorithm [28.6]

28.3.3 Determinization

We now describe the generic determinization algorithm for weighted automata that we used informally when working through the example in Sect. 28.2.4. This algorithm is a generalization of the classical subset construction for unweighted nondeterministic finite automata (NFAs). The determinization of unweighted or weighted finite-state transducers can both be viewed as special instances of the generic algorithm presented here but, for simplicity, we will focus on the weighted acceptor case.

A weighted automaton is *deterministic* (also called *subsequential*) if it has a unique initial state and if no two transitions leaving any state share the same input label. The *determinization* algorithm we now present applies to weighted automata over a cancellative weakly left-divisible semiring that satisfies a mild technical condition. If $x \in P(I, Q)$, then $w[P(I, x, Q)] \neq \overline{0}$, which is satisfied by trim automata over the tropical semiring or any other zero-sum-free semiring. Figure 28.10 gives pseudocode for the algorithm.

A *weighted subset* p of Q is a set of pairs $(q, x) \in Q \times \mathbb{K}$. $Q[p]$ is the set of states q in p, $E[Q[p]]$ is the set of transitions leaving those states, and $i[E[Q[p]]]$ is the set of input labels of those transitions.

The states of the result automaton are weighted subsets of the states of the original automaton. A state r of the result automaton that can be reached from the start state by path π is the weighted set of pairs $(q, x) \in Q \times \mathbb{K}$ such that q can be reached from an initial state of the original automaton by a path σ with $i[\sigma] = i[\pi]$ and $\lambda[p[\sigma]] \otimes w[\sigma] = \lambda[p[\pi]] \otimes w[\pi] \otimes x$. Thus, x can be viewed as the *residual* weight at state q. The algorithm takes as input a weighted automaton $A = (\mathcal{A}, Q, I, F, E, \lambda, \rho)$ and, when it terminates, yields an equivalent deterministic weighted automaton $A' = (\mathcal{A}, Q', I', F', E', \lambda', \rho')$.

The algorithm uses a queue S containing the set of states of the resulting automaton A', yet to be examined.

The sets Q', I', F', and E' are initially empty. The queue discipline for S can be chosen arbitrarily and does not affect the termination of the algorithm. The initial state set of A' is $I' = \{i'\}$ where i' is the weighted set of the initial states of A with the respective initial weights. Its initial weight is $\overline{1}$ (lines 1–2). S originally contains only the subset I' (line 3). Each time through the loop in lines 4–16 a new weighted subset p' is dequeued from S (lines 5–6). For each x labeling at least one of the transitions leaving a state p in the weighted subset p', a new transition with input label x is constructed. The weight w' associated to that transition is the sum of the weights of all transitions in $E[Q[p']]$ labeled with x pre-$\otimes$-multiplied by the residual weight v at each state p (line 8). The destination state of the transition is the subset containing all the states q reached by transitions in $E[Q[p']]$ labeled with x. The weight of each state q of the subset is obtained by taking the $\oplus$-sum of the residual weights of the states p $\otimes$-times the weight of the transition from p leading to q and by *dividing* that by w'. The new subset q' is inserted into the queue S when it is a new state (line 16). If any of the states in the subset q' is final, q' is made a final state and its final weight is obtained by summing the final weights of all the final states in q', pre-$\otimes$-multiplied by their residual weight v (line 14–15).

The worst-case complexity of determinization is exponential even in the unweighted case. However, in many practical cases such as for weighted automata used in large-vocabulary speech recognition, this blow-up does not occur. It is also important to notice that, just like composition, determinization has a natural lazy implementation in which only the transitions required by an application are expanded in the result automaton.

Unlike in the unweighted case, determinization does not halt on all input weighted automata. We say that a weighted automaton A is *determinizable* if the determinization algorithm halts for the input A. With a determinizable input, the algorithm outputs an equivalent deterministic weighted automaton.

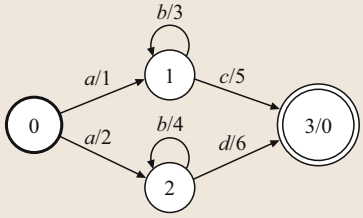


Fig. 28.11 Nondeterminizable weighted automaton over the tropical semiring. States 1 and 2 are non-twin siblings

The *twins property* for weighted automata characterizes determinizable weighted automata under some general conditions [28.6]. Let A be a weighted automaton over a weakly left-divisible semiring $\mathbb{K}$. Two states q and q' of A are said to be *siblings* if there are strings x and y in $\mathcal{A}^*$ such that both q and q' can be reached from I by paths labeled with x and there is a cycle at q and a cycle at q' both labeled with y. When $\mathbb{K}$ is a commutative and cancellative semiring, two sibling states are said to be *twins* when, for every string y,

$$w[P(q, y, q)] = w[P(q', y, q')] \,. \tag{28.14}$$

A has *the twins property* if any two sibling states of A are twins. Figure 28.11 shows a weighted automaton over the tropical semiring that does not have the twins property: states 1 and 2 can be reached by paths labeled with a from the initial state and have cycles with the same label b, but the weights of these cycles (3 and 4) are different.

The following theorems relate the twins property and weighted determinization [28.6].

Theorem 28.1
Let A be a weighted automaton over the tropical semiring. If A has the twins property, then A is determinizable.

A weighted automaton is said to be *unambiguous* if, for any $x \in \Sigma^*$, it admits at most one accepting path labeled with x.

Theorem 28.2
Let A be a trim unambiguous weighted automaton over the tropical semiring. Then A is determinizable iff it has the twins property.

There is an efficient algorithm for testing the twins property for weighted automata [28.25]. Note that any acyclic weighted automaton over a zero-sum-free semiring has the twins property and is determinizable.

The *pre-determinization* algorithm can be used to make an arbitrary weighted transducer determinizable over the tropical semiring by inserting transitions labeled with special symbols [28.26]. The algorithm makes use of a general twins property [28.25] to insert new transitions when needed to guarantee that the resulting transducer has the twins property and thus is determinizable.

28.3.4 Weight Pushing

As discussed in Sect. 28.2.5, *weight pushing* is necessary in weighted minimization, and is also very useful to improve search. Weight pushing can also be used to test the equivalence of two weighted automata. Weight pushing is possible because the choice of the distribution of the total weight along each successful path of a weighted automaton does not affect the total weight of each successful path, and therefore preserves the definition of the automaton as a weighted set (weighted relation for a transducer).

Let A be a weighted automaton over a zero-sum-free and weakly left-divisible semiring $\mathbb{K}$. For any state $q \in Q$, assume that the following sum is well defined and in $\mathbb{K}$:

$$d[q] = \bigoplus_{\pi \in P(q,F)} (w[\pi] \otimes \rho[n[\pi]]) \,. \tag{28.15}$$

The value $d[q]$ is the *shortest distance* from q to F [28.20]. This is well defined for all $q \in Q$ when $\mathbb{K}$ is a closed semiring. The weight-pushing algorithm consists of computing each shortest-distance $d[q]$ and of *reweighting* the transition weights, initial weights and final weights in the following way:

$$\begin{aligned} w[e] &\leftarrow d[p[e]]^{-1} \otimes w[e] \otimes d[n[e]] \text{ if } d[p[e]] \neq \overline{0} \\ \lambda[i] &\leftarrow \lambda[i] \otimes d[i] \,, \\ \rho[f] &\leftarrow d[f]^{-1} \otimes \rho[f] \text{ if } d[f] \neq \overline{0} \,, \end{aligned} \tag{28.16}$$

where e is any transition, i any initial state, and f any final state. Each of these operations can be done in constant time. Therefore, reweighting can be done in linear time $O(T_{\otimes}|A|)$ where $T_{\otimes}$ denotes the worst cost of an $\otimes$-operation. The complexity of the shortest-distances computation depends on the semiring [28.20]. For the tropical semiring, $d[q]$ can be computed using a standard shortest-distance algorithm. The complexity is linear for acyclic automata, $O[|Q| + (T_{\oplus} + T_{\otimes})|E|]$, where $T_{\oplus}$ denotes the worst cost of an $\oplus$-operation. For general weighted automata over the tropical semiring, the complexity is $O(|E| + |Q| \log |Q|)$.

For semirings like the probability semiring, a generalization of the Floyd–Warshall algorithm for computing all-pairs shortest distances can be used [28.20]. Its complexity is $\Theta(|Q|^3(T_{\oplus} + T_{\otimes} + T_*))$ where T_* denotes the worst cost of the closure operation. The space complexity of these algorithms is $\Theta(|Q|^2)$. Therefore, the Floyd–Warshall algorithm is impractical for automata with several hundred million states or transitions, which arise in large-vocabulary ASR. An approximate version of a generic single-source shortest-distance algorithm can be used instead to compute $d[q]$ efficiently [28.20].

Speaking informally, the algorithm pushes the weight on each path as much as possible towards the initial states. Figure 28.5a,b show weight pushing on the tropical semiring, while Fig. 28.12 shows weight pushing for the same automaton but on the probability semiring.

Note that if $d[q] = \overline{0}$, then, since $\mathbb{K}$ is zero-sum-free, the weight of all paths from q to F is $\overline{0}$. Let A be a weighted automaton over the semiring $\mathbb{K}$. Assume that $\mathbb{K}$ is closed and that the shortest distances $d[q]$ are all well defined and in $\mathbb{K} - \{\overline{0}\}$. In both cases, we can use the distributivity over the infinite sums defining shortest distances. Let e' (π') denote the transition e (path π) after application of the weight-pushing algorithm. e' (π') differs from e (resp. π) only by its weight. Let λ' denote the new initial weight function, and ρ' the new final weight function. Then, the following proposition holds [28.6, 27].

Proposition 28.1

Let $B = (\mathcal{A}, Q, I, F, E', \lambda', \rho')$ be the result of the weight-pushing algorithm applied to the weighted automaton A, then

1. the weight of a successful path π is unchanged after weight pushing:

$$\lambda'[p[\pi']] \otimes w[\pi'] \otimes \rho'[n[\pi']] = \lambda[p[\pi]] \otimes w[\pi] \otimes \rho[n[\pi]] \,. \tag{28.17}$$

2. the weighted automaton B is *stochastic*, that is,

$$\forall q \in Q, \bigoplus_{e' \in E'[q]} w[e'] = \overline{1} \,. \tag{28.18}$$

These two properties of weight pushing are illustrated by Figs. 28.5a,b and 28.12: the total weight of a successful path is unchanged after pushing; at each

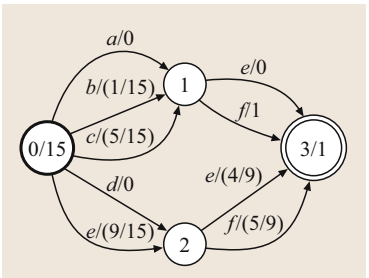


Fig. 28.12 Weighted automaton C obtained from A of Fig. 28.5a by weight pushing in the probability semiring

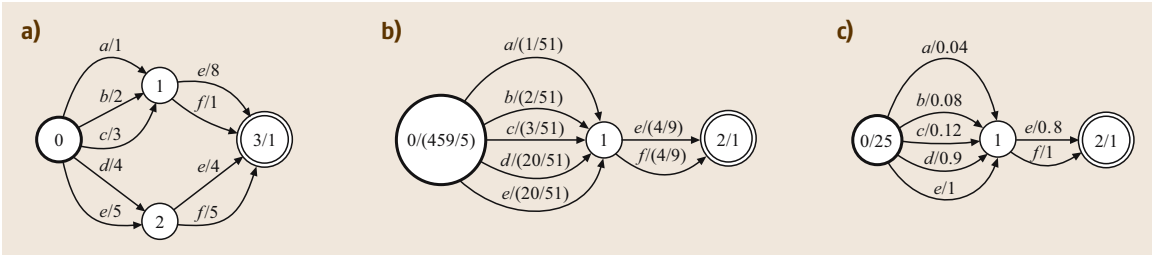


Fig. 28.13a–c Minimization of weighted automata. **(a)** Weighted automaton A' over the probability semiring. **(b)** Minimal weighted automaton B' equivalent to A'. **(c)** Minimal weighted automaton C' equivalent to A'

state of the weighted automaton of Fig. 28.5b, the minimum weight of the outgoing transitions is 0, and at each state of the weighted automaton of Fig. 28.12, the weights of outgoing transitions sum to 1.

28.3.5 Minimization

Finally, we discuss in more detail the minimization algorithm introduced in Sect. 28.2.5. A deterministic weighted automaton is said to be *minimal* if there is no other deterministic weighted automaton with a smaller number of states that represents the same mapping from strings to weights. It can be shown that the minimal deterministic weighted automaton also has the minimal number of transitions among all equivalent deterministic weighted automata [28.6].

Two states of a deterministic weighted automaton are said to be *equivalent* if exactly the same set of strings label the paths from these states to a final state, and the total weight of the paths for each string, including the final weight of the final state, is the same. Thus, two equivalent states of a deterministic weighted automaton can be merged without affecting the function realized by that automaton. A weighted automaton is minimal when it is not possible to create two distinct equivalent states after any pushing of the weights along its paths.

As outlined in Sect. 28.2.5, the general minimization algorithm for weighted automata consists of first applying the weight-pushing algorithm to normalize the distribution of the weights along the paths of the input automaton, and then of treating each pair (label, weight) as a single label and applying classical (unweighted) automata minimization [28.6]. The minimization of both unweighted and weighted finite-state transducers can also be viewed as instances of the algorithm presented here, but, for simplicity, we will not discuss that further here. The following theorem holds [28.6].

Theorem 28.3

Let A be a deterministic weighted automaton over a semiring $\mathbb{K}$. Assume that the conditions of application of the weight pushing algorithm hold. Then the execution of the following steps:

1. weight pushing,
2. (unweighted) automata minimization,

lead to a minimal weighted automaton equivalent to A.

The complexity of automata minimization is linear in the case of acyclic automata $O(|Q|+|E|)$ and is $O(|E|\log|Q|)$ in the general case. In view of the complexity results of the previous section, for the tropical semiring, the time complexity of the weighted minimization algorithm is linear $O(|Q|+|E|)$ in the acyclic case and $O(|E|\log|Q|)$ in the general case.

Figure 28.5 illustrates the algorithm in the tropical semiring. Automaton A cannot be further minimized using the classical unweighted automata minimization since no two states are equivalent in that machine. After weight pushing, automaton B has two states, 1 and 2, which can be merged by unweighted automaton minimization.

Figure 28.13 illustrates the minimization of an automaton defined over the probability semiring. Unlike the unweighted case, a minimal weighted automaton is not unique, but all minimal weighted automata have the same graph topology, and only differ in the weight distribution along each path. The weighted automata B' and C' are both minimal and equivalent to A'. B' is obtained from A' using the algorithm described above in the probability semiring and it is thus a stochastic weighted automaton in the probability semiring.

For a deterministic weighted automaton, the $\oplus$ operation can be arbitrarily chosen without affecting the

mapping from strings to weights defined by the automaton, because a deterministic weighted automaton has at most one path labeled by any given string. Thus, in the algorithm described in theorem 28.3, the weight pushing step can be executed in any semiring $\mathbb{K}'$ whose multiplicative operation matches that of $\mathbb{K}$. The minimal weighted automata obtained by pushing the weights in $\mathbb{K}'$ is also minimal in $\mathbb{K}$, since it can be interpreted as a (deterministic) weighted automaton over $\mathbb{K}$.

In particular, A' can be interpreted as a weighted automaton over the semiring $(\mathbb{R}_+, \max, \times, 0, 1)$. The application of the weighted minimization algorithm to A' in this semiring leads to the minimal weighted automaton C' of Fig. 28.13c. C' is also a *stochastic* weighted automaton in the sense that, at any state, the maximum weight of all outgoing transitions is one.

In the particular case of a weighted automaton over the probability semiring, it may be preferable to use weight pushing in the $(\max, \times)$-semiring since the complexity of the algorithm is then equivalent to that of classical single-source shortest-paths algorithms. The corresponding algorithm is a special instance of a generic shortest-distance algorithm [28.20]. Using the $(\max \times)$-semiring assumes the resulting distribution of weights along paths is not important. As discussed in the next section, the weight distribution that results from pushing in the $(+, \times)$ semiring has advantages when the resulting automaton is used in a pruned search.

28.4 Applications to Speech Recognition

We now describe the details of the application of weighted finite-state transducer representations and algorithms to speech recognition as introduced in Sect. 28.2.

28.4.1 Speech Recognition Transducers

As described in Sect. 28.2, we will represent various models in speech recognition as weighted finite-state transducers. Four principal models are the word-level grammar G, the pronunciation lexicon L, the context-dependency transducer C, and the HMM transducer H. We will now discuss the construction of each these transducers. Since these will be combined by composition and optimized by determinization, we ensure they are efficient to compose and allow weighted determinization.

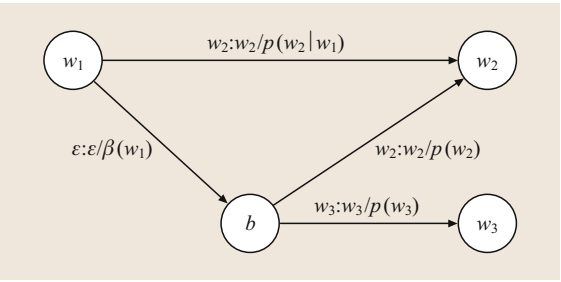


Fig. 28.14 Word bigram transducer model: seen bigram w_1w_2 represented as a w_2-transition from the state w_1 to state w_2; unseen bigram w_1w_3 represented as an ϵ-transition from state w_1 to backoff state b and as a w_3 transition from state b to state w_3

The word-level grammar G, whether handcrafted or learnt from data, is typically a finite-state model in speech recognition. Handcrafted finite-state models can be specified by regular expressions, rules or directly as automata. Stochastic n-gram models, common in large-vocabulary speech recognition, can be represented compactly by finite-state models. For example, a bigram grammar has a state for every word w_i and a transition from state w_1 to state w_2 for every bigram w_1w_2 that is seen in the training corpus. The transition is labeled with w_2 and has weight $-\log(\hat{p}(w_2|w_1))$, the negative log of the estimated transition probability. The weight of a bigram w_1w_3 that is not seen in the training data can be estimated, for example, by backing-off to the unigram. That is, it has weight $-\log(\beta(w_1)\,\hat{p}(w_3))$, where $\hat{p}(w_3)$ is the estimated w_3 unigram probability and $\beta(w_1)$ is the w_1 back-off weight [28.28]. The unseen bigram could be represented as a transition from state w_1 to w_3 in the bigram automaton just as a seen bigram. However, this would result in $O(|V|^2)$ transitions in the automaton, where $|V|$ is the vocabulary size. A simple approximation, with the introduction of a *back-off* state b, avoids this. In this model, an unseen w_1w_3 bigram is represented as two transitions: an ε-transition from state w_1 to state b with weight $-\log(\beta(w_1))$ and a transition from state b to state w_3 with label w_3 and weight $-\log(\hat{p}(w_3))$. This configuration is depicted in Fig. 28.14. This is an approximation since seen bigrams may also be read as backed-off unigrams. However, since the seen bigram typically has higher probability than its backed-off unigram, it is usually a good approx-

imation. A similar construction is used for higher-order n-grams.

These grammars present no particular issues for composition. However, the back-off ε-transitions introduce nondeterminism in the n-gram model. If fully determinized without ε-transitions, $O(|V|^2)$ transitions would result. However, we can treat the back-off ε labels as regular symbols during determinization, avoiding the explosion in the number of transitions.

As described in Sect. 28.2, we represent the pronunciation lexicon L as the Kleene closure of the union of individual word pronunciations, as in Fig. 28.2b. In order for this transducer to efficiently compose with G, the output (word) labels must be placed on the initial transitions of the words; other locations would lead to delays in the composition matching, which could consume significant time and space.

In general, transducer L is not determinizable. This is clear in the presence of homophones. But, even without homophones, it may not be determinizable because the first word of the output string might not be known before the entire phone string is scanned. Such unbounded output delays make L nondeterminizable.

To make it possible to determinize L, we introduce an auxiliary phone symbol, denoted $\#_0$, marking the end of the phonetic transcription of each word. Other auxiliary symbols $\#_1 \ldots \#_{k-1}$ are used when necessary to distinguish homophones, as in the following example:

`r eh d` $\#_0$ *read*

`r eh d` $\#_1$ *red* .

At most P auxiliary phones are needed, where P is the maximum degree of homophony. The pronunciation dictionary transducer with these auxiliary symbols added

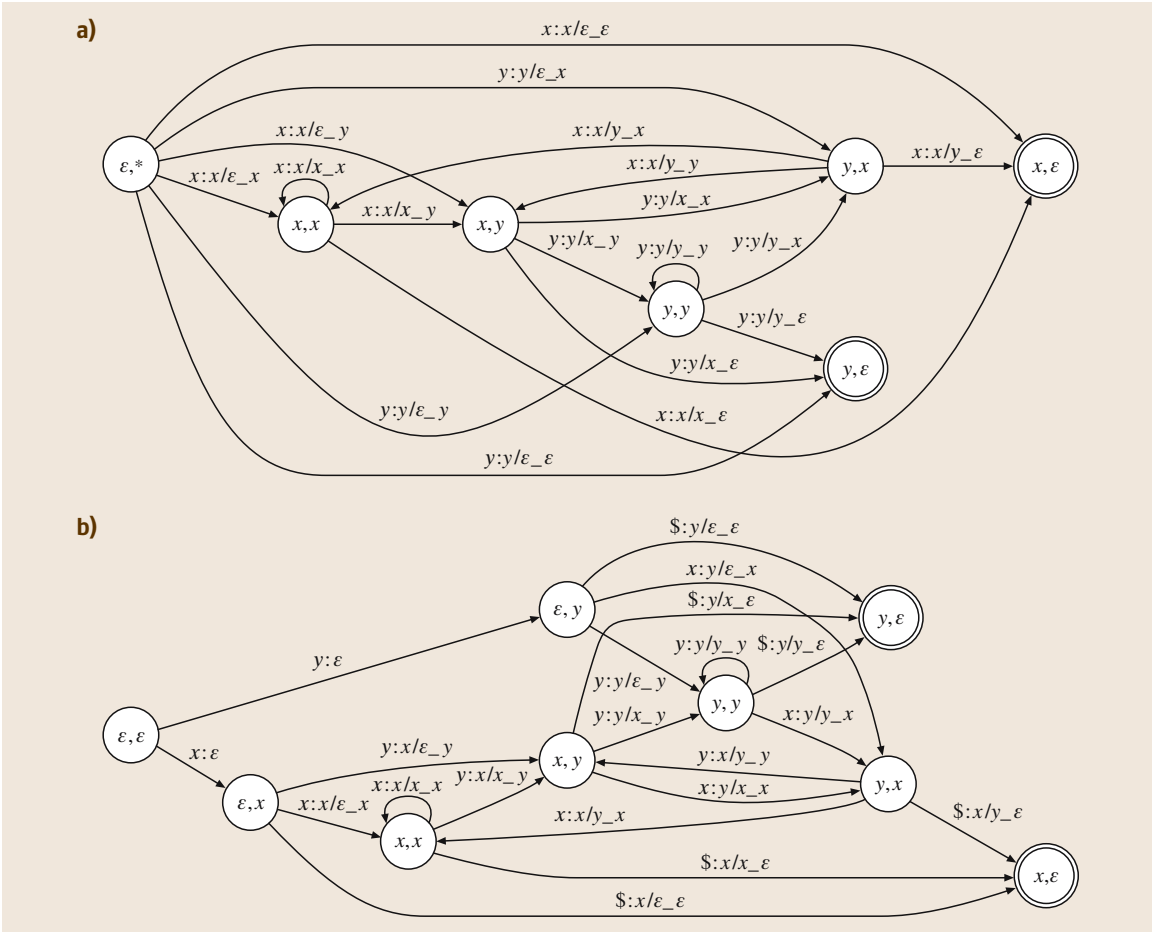


Fig. 28.15a,b Context-dependent triphone transducers: **(a)** nondeterministic, **(b)** deterministic

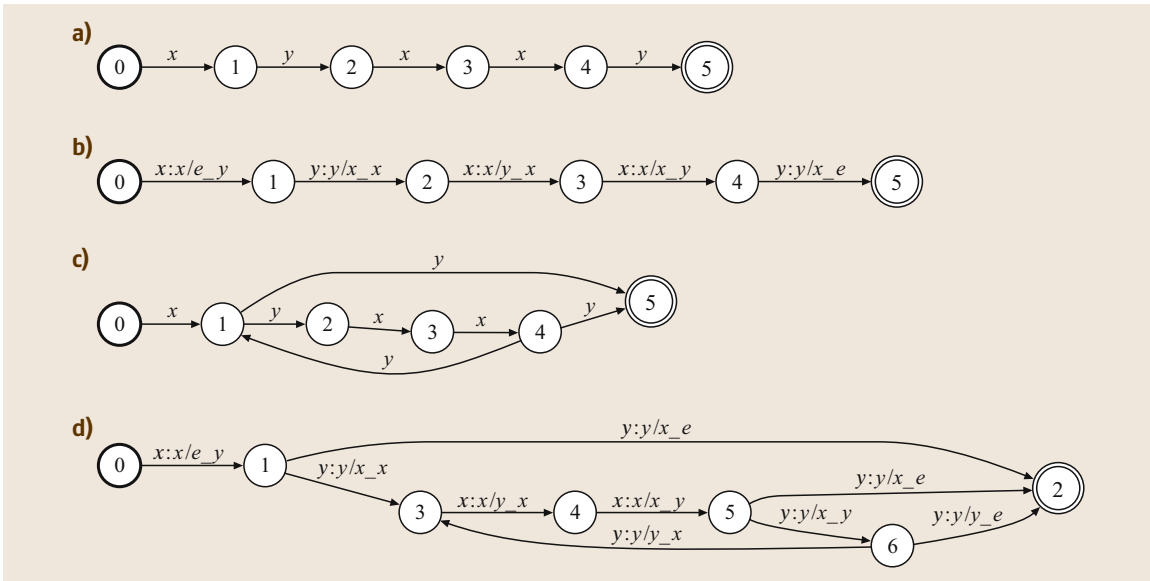


Fig. 28.16a–d Context-dependent composition examples: **(a)** context-independent 'string', **(b)** context dependency applied to 1, **(c)** context-independent automaton, **(d)** context dependency applied to **(c)**

is denoted by $\tilde{L}$. *Allauzen* et al. [28.29] describe more-general alternatives to the direct construction of $\tilde{L}$. In that work, as long as L correctly defines the pronunciation transduction, it can be transformed algorithmically to something quite similar to $\tilde{L}$, regardless of the initial disposition of the output labels or the presence of homophony.

As introduced in Sect. 28.2, we can represent the mapping from context-independent phones to context-dependent units with a finite-state transducer, with Fig. 28.6 giving a transition of that transducer. Figure 28.15 gives complete context-dependency transducers where just two hypothetical phones x and y are shown for simplicity. The transducer in Fig. 28.15a is nondeterministic, while the one in Fig. 28.15b is deterministic. For illustration purposes, we will describe the nondeterministic version since it is somewhat simpler. As in Sect. 28.2, we denote the context-dependent units as *phone/left context_right context*. Each state in Fig. 28.15a encodes the knowledge of the previous and next phones. State labels in the figure are pairs (a, b) of the past a and the future b, with ε representing the start or end of a phone string and $*$ an unspecified future. For instance, it is easy to see that the phone string xyx is mapped by the transducer to $x/\varepsilon\_y\ y/x\_x\ x/y\_\varepsilon$ via the unique state sequence $(\varepsilon, *)(x, y)(y, x)(x, \varepsilon)$. More generally, when there are n context-independent phones, this triphonic construction gives a transducer with $O(n^2)$ states and $O(n^3)$ transitions. A tetraphonic construction would give a transducer with $O(n^3)$ states and $O(n^4)$ transitions.

The following simple example shows the use of this context-dependency transducer. A context-independent string can be represented by the obvious single-path acceptor as in Fig. 28.16a. This can then be composed with the context-dependency transducer in Fig. 28.15. Before composition, we promote the acceptor in Fig. 28.16a to the corresponding transducer with identical input and output labels. The result is the transducer in Fig. 28.16b, which has a single path labeled with the context-independent labels on the input side and the corresponding context-dependent labels on the output side.

The context-dependency transducer can be composed with more-complex transducers than the trivial one in Fig. 28.16a. For example, composing the context-dependency transducer with the transducer in Fig. 28.16c results in the transducer in Fig. 28.16d. By definition of relational composition, this must correctly replace the context-independent units with the appropriate context-dependent units on all of its paths. Therefore, composition provides a convenient and general mechanism for applying context dependency to ASR transducers.

The nondeterminism of the transducer in Fig. 28.15a introduces a single symbol-matching delay in the com-

position with the lexicon. The deterministic transducer in Fig. 28.15b composes without a matching delay, which makes it the better choice in applications. However, it introduces a single-phone shift between a context-independent phone and its corresponding context-dependent unit in the result. This shift requires the introduction of a final *subsequential* symbol \$ to pad out the context. In practice, this might be mapped to a silence phone or an ε-transition.

If we let C represent a context-dependency transducer from context-dependent phones to context-independent phones, then

$$C \circ L \circ G$$

gives a transducer that maps from context-dependent phones to word strings restricted to the grammar G. Note that C is the inverse of a transducer such as in Fig. 28.15; that is, the input and output labels have been exchanged on all transitions. For notational convenience, we adopt this form of the context-dependency transducer when we use it in recognition cascades.

For correctness, the context-dependency transducer C must also accept all paths containing the auxiliary symbols added to $\tilde{L}$ to make it determinizable. For determinizations at the context-dependent phone level and distribution level, each auxiliary phone must be mapped to a distinct context-dependent-level symbol. Thus, self-loops are added at each state of C to map each auxiliary phone to a new auxiliary context-dependent phone. The augmented context-dependency transducer is denoted by $\tilde{C}$.

As we did for the pronunciation lexicon, we can represent the HMM set as H, the closure of the union of the individual HMMs (see Fig. 28.16c). Note that we do not explicitly represent the HMM-state self-loops in H. Instead, we simulate those in the run-time decoder. With H in hand,

$$H \circ C \circ L \circ G$$

gives a transducer that maps from distributions to word strings restricted to G.

Each auxiliary context-dependent phone in $\tilde{C}$ must be mapped to a new distinct distribution name. Self-loops are added at the initial state of H with auxiliary distribution name input labels and auxiliary context-dependent phone output labels to allow for this mapping. The modified HMM model is denoted by $\tilde{H}$.

We thus can use composition to combine all levels of our ASR transducers into an integrated transducer in a convenient, efficient, and general manner. When these automata are statically provided, we can apply the optimizations discussed in the next section to reduce decoding time and space requirements. If the transducer needs to be modified dynamically, for example by adding the results of a database lookup to the lexicon and grammar in an extended dialogue, we adopt a hybrid approach that optimizes the fixed parts of the transducer and uses lazy composition to combine them with the dynamic portions during recognition [28.30, 31].

28.4.2 Transducer Standardization

To optimize an integrated transducer, we use three additional steps: (a) determinization, (b) minimization, and (c) factoring.

Determinization

We use weighted transducer determinization at each step of the composition of each pair of transducers. The main purpose of determinization is to eliminate redundant paths in the composed transducer, thereby substantially reducing recognition time. In addition, its use in intermediate steps of the construction also helps to improve the efficiency of composition and to reduce transducer size.

First, $\tilde{L}$ is composed with G and determinized, yielding $\det(\tilde{L} \circ G)$. The benefit of this determinization is the reduction of the number of alternative transitions at each state to at most the number of distinct phones at that state, while the original transducer may have as many as V outgoing transitions at some states, where V is the vocabulary size. For large tasks in which the vocabulary has 10^5 to 10^6 words, the advantages of this optimization are clear.

$\tilde{C}$ is then composed with the resulting transducer and determinized. Similarly, $\tilde{H}$ is composed with the context-dependent transducer and determinized. This last determinization increases sharing among HMM models that start with the same distributions. At each state of the resulting integrated transducer, there is at most one outgoing transition labeled with any given distribution name, reducing the recognition time even more.

In a final step, we use the erasing operation π_ε that replace the auxiliary distribution symbols by ε's. The complete sequence of operations is summarized by the following construction formula:

$$N = \pi_\varepsilon(\det(\tilde{H} \circ \det(\tilde{C} \circ \det(\tilde{L} \circ G)))) \,, \qquad (28.19)$$

where parentheses indicate the order in which the operations are performed. The result N is an integrated recognition transducer that can be constructed even in

very large-vocabulary tasks and leads to a substantial reduction in recognition time, as the experimental results below show.

Minimization

Once we have determinized the integrated transducer, we can reduce it further by minimization. The auxiliary symbols are left in place, the minimization algorithm is applied, and then the auxiliary symbols are removed:

$$N = \pi_\varepsilon(\min(\det(\tilde{H} \circ \det(\tilde{C} \circ \det(\tilde{L} \circ G))))) \,. \quad (28.20)$$

Weighted minimization can be used in different semirings. Both minimization in the tropical semiring and minimization in the log semiring can be used in this context. It is not hard to prove that the results of these two minimizations have exactly the same number of states and transitions and only differ in how weight is distributed along paths. The difference in weights arises from differences in the definition of the weight-pushing operation for different semirings.

Weight pushing in the log semiring has a very large, beneficial impact on the pruning efficacy of a standard

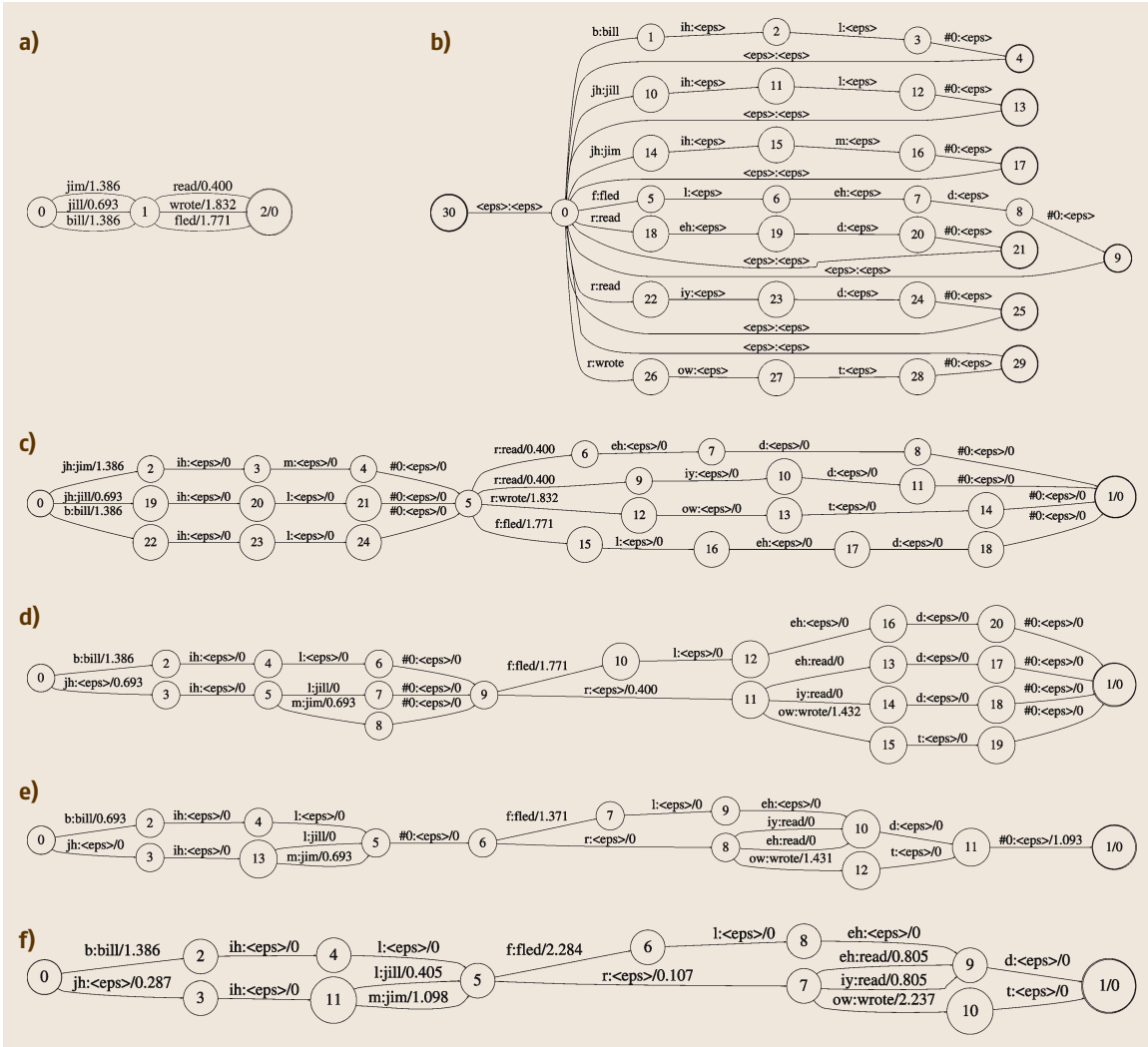


Fig. 28.17a–f Recognition transducer construction: **(a)** grammar G, **(b)** lexicon $\tilde{L}$, **(c)** $\tilde{L} \circ G$, **(d)** $\det(\tilde{L} \circ G)$, **(e)** $\min_{\text{tropical}}(\det(\tilde{L} \circ G))$, **(f)** $\min_{\log}(\det(\tilde{L} \circ G))$

Viterbi beam search. In contrast, weight pushing in the tropical semiring, which is based on lowest weights between paths described earlier, produces a transducer that may slow down beam-pruned Viterbi decoding many fold.

To push weights in the log semiring instead of the tropical semiring, the potential function is the $-\log$ of the total probability of paths from each state to the super-final state rather than the lowest weight from the state to the super-final state. In other words, the transducer is pushed in terms of probabilities along all future paths from a given state rather than the highest probability over the single best path. By using $-\log$ probability pushing, we preserve a desirable property of the language model, namely that the weights of the transitions leaving each state are normalized as in a probabilistic automaton [28.32]. We have observed that probability pushing makes pruning more effective [28.21], and conjecture that this is because the acoustic likelihoods and the transducer probabilities are now *synchronized* to obtain the optimal likelihood ratio test for deciding whether to prune. We further conjecture that this reweighting is the best possible for pruning. A proof of these conjectures will require a careful mathematical analysis of pruning.

We have thus *standardized* the integrated transducer in our construction – it is the *unique* deterministic, minimal transducer for which the weights for all transitions leaving any state sum to 1 in probability, up to state relabeling. If one accepts that these are desirable properties of an integrated decoding transducer, then our methods obtain the *optimal* solution among all integrated transducers.

Figure 28.17 illustrates the steps in this construction. For simplicity, we consider a small toy grammar and show the construction only down to the context-independent phone level. Figure 28.17a shows the toy grammar G and Fig. 28.17b shows the lexicon $\tilde{L}$. Note the word labels on the lexicon are on the initial transitions and that disambiguating auxiliary symbols have been added at the word ends. Figure 28.17c shows their composition $\tilde{L} \circ G$. Figure 28.17d shows the resulting determinization, $\det(\tilde{L} \circ G)$; observe how phone redundancy is removed. Figure 28.17e,f shows the minimization step, $\min(\det(\tilde{L} \circ G))$; identical futures are combined. In Fig. 28.17e, the minimization uses weight pushing over the tropical semiring, while in Fig. 28.17f, the log semiring is used.

Factoring

For efficiency reasons, our decoder has a separate representation for variable-length left-to-right HMMs, which we will call the *HMM specification*. The integrated transducer of the previous section does not take good advantage of this since, having combined the HMMs into the recognition transducer proper, the HMM specification consists of trivial one-state HMMs. However, by suitably *factoring* the integrated transducer, we can again take good advantage of this feature.

A path whose states other than the first and last have at most one outgoing and one incoming transition is called a *chain*. The integrated recognition transducer just described may contain many chains after the composition with $\tilde{H}$, and after determinization. As mentioned before, we do not explicitly represent the HMM-state self-loops but simulate them in the run-time decoder. The set of all chains in N is denoted by $\text{chain}(N)$.

The input labels of N name one-state HMMs. We can replace the input of each length-n chain in N by a single label naming an n-state HMM. The same label is used for all chains with the same input string. The result of that replacement is a more-compact transducer denoted by F. The factoring operation on N leads to the following decomposition:

$$N = H' \circ F \,, \tag{28.21}$$

where H' is a transducer mapping variable-length left-to-right HMM state distribution names to n-state HMMs. Since H' can be separately represented in the decoder's HMM specification, the actual recognition transducer is just F.

Chain inputs are in fact replaced by a single label only when this helps to reduce the size of the transducer. This can be measured by defining the *gain* of the replacement of an input string σ of a chain by:

$$G(\sigma) = \sum_{\pi \in \text{chain}(N), i[\pi]=\sigma} |\sigma| - |o[\pi]| - 1 \,, \tag{28.22}$$

where $|\sigma|$ denotes the length of the string σ, $i[\pi]$ the input label, and $o[\pi]$ the output label of a path π. The replacement of a string σ helps to reduce the size of the transducer if $G(\sigma) > 0$.

Our implementation of the factoring algorithm allows one to specify the maximum number r of replacements done (the r chains with the highest gain are replaced), as well as the maximum length of the chains that are factored.

Factoring does not affect recognition time. It can however significantly reduce the size of the recognition transducer. We believe that even better factoring methods may be found in the future.

Table 28.2 Size of the first-pass recognition transducers in the NAB 40 000-word vocabulary task

Transducer	States	Transitions
G	1 339 664	3 926 010
$L \circ G$	8 606 729	11 406 721
$\det(L \circ G)$	7 082 404	9 836 629
$C \circ \det(L \circ G)$	7 273 035	10 201 269
$\det(H \circ C \circ L \circ G)$	18 317 359	21 237 992
F	3 188 274	6 108 907
$\min(F)$	2 616 948	5 497 952

Table 28.3 **(a)** Recognition speed of the first-pass transducers in the NAB 40, 000-word vocabulary task at 83% word accuracy. **(b)** Recognition speed of the second-pass transducers in the NAB 160 000-word vocabulary task at 88% word accuracy

a) Transducer	× real time
$C \circ L \circ G$	12.5
$C \circ \det(L \circ G)$	1.2
$\det(H \circ C \circ L \circ G)$	1.0
$\min(F)$	0.7
b) Transducer	× real time
$C \circ L \circ G$	0.18
$C \circ \det(L \circ G)$	0.13
$C \circ \min(\det(L \circ G))$	0.02

Experimental Results – First-Pass Transducers

We have used the techniques discussed in the previous sections to build many recognizers. To illustrate the effectiveness of the techniques and explain some practical details, we discuss here an integrated, optimized recognition transducer for a 40 000-word vocabulary North American Business News (NAB) task. The following models are used:

- An acoustic model of 7208 distinct HMM states, each with an emission mixture distribution of up to 12 Gaussians.
- Triphonic context-dependency transducer C with 1525 states and 80 225 transitions.
- A 40 000-word pronunciation dictionary L with an average of 1.056 pronunciations per word and an out-of-vocabulary rate of 2.3% on the NAB Eval '95 test set.
- A trigram language model G with 3 926 010 transitions built by Katz's back-off method with frequency cutoffs of 2 for bigrams and 4 for trigrams, shrunk with an epsilon of 40 using the method of [28.33], which retained all the unigrams, 22.3% of the bigrams, and 19.1% of the trigrams. The perplexity on the NAB Eval '95 test set is 164.4 (142.1 before shrinking).

We applied the transducer optimization steps as described in the previous section except that we applied the minimization and weight pushing after factoring the transducer. Table 28.2 gives the size of the intermediate and final transducers.

Observe that the factored transducer $\min(F)$ has only about 40% more transitions than G. The HMM specifi-

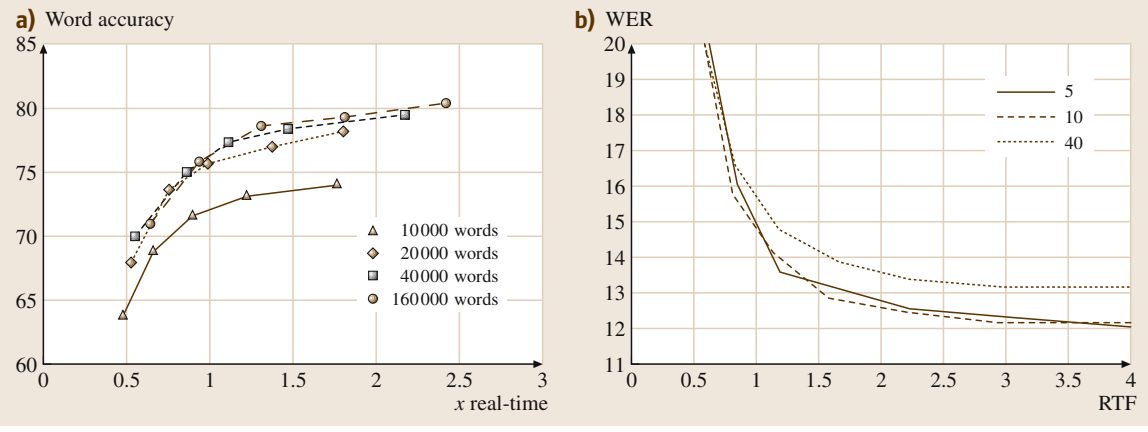


Fig. 28.18 **(a)** Effect of vocabulary size: NAB bigram recognition results for different vocabulary sizes (LG optimized only). **(b)** NAB Eval 1995 recognition results for the Seymore and Rosenfeld shrink factors of 5, 10, and 40 (thanks to RWTH; uses RWTH acoustic models)

cation H' consists of 430 676 HMMs with an average of 7.2 states per HMM. It occupies only about 10% of the memory of $\min(F)$ in the decoder (due to the compact representation possible from its specialized topology). Thus, the overall memory reduction from factoring is substantial.

We used these transducers in a simple, general-purpose, one-pass Viterbi decoder applied to the DARPA NAB Eval '95 test set. Table 28.3a shows the recognition speed on a Compaq Alpha 21264 processor for the various optimizations, where the word accuracy has been fixed at 83.0%. We see that the fully optimized recognition transducer, $\min(F)$, substantially speeds up recognition.

To obtain improved accuracy, we might widen the decoder beam, use a larger vocabulary, or use a less-shrunken language model. These models have an asymptotic wide-beam accuracy of 85.3%. Figure 28.18a shows the affect of vocabulary size (with a bigram LM and optimization only to the $L \circ G$ level). We see that beyond 40 000 words, there is little benefit to increasing the vocabulary either in real-time performance or asymptotically. Figure 28.18b shows the affect of the language model shrinking parameter. These curves were produced by Stephan Kanthak of RWTH using our transducer construction, but RWTH's acoustic models, as part of a comparison with lexical tree methods [28.34]. As we can see, decreasing the shrink parameter from 40 as used above to 10 has a significant effect, while further reducing it to 5 has very little effect. An alternative to using a larger LM is to use a two-pass system to obtain improved accuracy, as described in the next section. This has the advantage tha tit allows quite compact shrunken bigram LMs in the first pass, while the second pass performs as well as the larger-model single-pass systems.

While our examples here have been on NAB, we have also applied these methods to Broadcast News [28.35], Switchboard, and various AT&T-specific large-vocabulary tasks [28.29]. In our experience, fully optimized and factored recognition transducers provide very fast decoding while often having substantially less than twice the number of transitions as their word-level grammars.

Experimental Results – Rescoring Transducers

The weighted transducer approach is also easily applied to multipass recognition. To illustrate this, we now show how to implement lattice rescoring for a 160 000-word vocabulary NAB task. The following models are used to build lattices in a first pass:

- An acoustic model of 5520 distinct HMM states, each with an emission mixture distribution of up to four Gaussians.
- A triphonic context-dependency transducer C with 1525 states and 80 225 transitions.
- A 160 000-word pronunciation dictionary L with an average of 1.056 pronunciations per word and an out-of-vocabulary rate of 0.8% on the NAB Eval '95 test set.
- A bigram language model G with 1 238 010 transitions built by Katz's back-off method with frequency cutoffs of 2 for bigrams. It is shrunk with an epsilon of 160 using the method of *Seymore* and *Rosenfeld* [28.33], which retained all the unigrams and 13.9% of the bigrams. Perplexity on the NAB Eval '95 test set is 309.9.

We used an efficient approximate lattice-generation method [28.36] to generate word lattices. These word lattices are then used as the grammar in a second rescoring pass. The following models are used in the second pass:

- An acoustic model of 7208 distinct HMM states, each with an emission mixture distribution of up to 12 Gaussians. The model is adapted to each speaker using a single full-matrix MLLR (maximum likelihood linear regression) transform [28.37].
- A triphonic context-dependency transducer C with 1525 states and 80 225 transitions.
- A 160 000-word stochastic, TIMIT-trained, multiple-pronunciation lexicon L [28.38].
- A 6-gram language model G with 40 383 635 transitions built by Katz's back-off method with frequency cutoffs of 1 for bigrams and trigrams, 2 for 4-grams, and 3 for 5-grams and 6-grams. It is shrunk with an epsilon of 5 using the method of Seymore and Rosenfeld, which retained all the unigrams, 34.6% of the bigrams, 13.6% of the trigrams, 19.5% of the 4-grams, 23.1% of the 5-grams, and 11.73% of the 6-grams. The perplexity on the NAB Eval '95 test set is 156.83.

We applied the transducer optimization steps described in the previous section but only to the level of $L \circ G$ (where G is each lattice). Table 28.3b shows the speed of second-pass recognition on a Compaq Alpha 21264 processor for these optimizations when

the word accuracy is fixed at 88.0% on the DARPA Eval '95 test set. The recognition speed excludes the offline transducer construction time. We see that the optimized recognition transducers again substantially speed up recognition. The median number of lattice states and arcs is reduced by $\sim 50\%$ by the optimizations.

28.5 Conclusion

We presented an overview of weighted finite-state transducer methods and their application to speech recognition. The methods are quite general, and can also be applied in other areas of speech and language processing, including information extraction, speech synthesis [28.39, 40], phonological and morphological analysis [28.41, 42], optical character recognition, biological sequence analysis, and other pattern-matching and string-processing applications [28.43], and image processing [28.44], to mention just some of the most active application areas.

References

28.1 F. Pereira, R. Wright: Finite-state approximation of phrase-structure grammars. In: *Finite-State Language Processing*, ed. by E. Roche, Y. Schabes (MIT Press, Cambridge, MA 1997) pp. 149–173

28.2 M-J Nederhof: Practical experiments with regular approximation of context-free languages, Computational Linguistics, 26(1) (2000)

28.3 M. Mohri, M.-J. Nederhof: Regular approximation of context-free grammars through transformation. In: *Robustness in Language and Speech Technology*, ed. by J.C. Junqua, G. van Noord (Kluwer Academic, The Netherlands 2001) pp. 153–163

28.4 M. Mohri, F. Pereira, M. Riley: The design principles of a weighted finite-state transducer library, Theoret. Comput. Sci. **231**, 17–32 (2000)

28.5 F. Pereira, M. Riley: *Finite State Language Processing, chapter Speech Recognition by Composition of Weighted Finite Automata* (MIT Press, Cambridge, MA 1997)

28.6 M. Mohri: Finite-state transducers in language and speech processing, Computational Linguistics 23(2) (1997)

28.7 M. Mohri, F. Pereira, Michael Riley: Weighted automata in text and speech processing, ECAI-96 Workshop, Budapest ECAI (1996)

28.8 M. Mohri, M. Riley: Network optimizations for large vocabulary speech recognition, Speech Commun. 25(3) (1998)

28.9 M. Mohri, M. Riley, D. Hindle, A. Ljolje, F. Pereira: full expansion of context-dependent networks in large vocabulary speech recognition, Procs. ICASSP (ICASSP '98), Seattle (1998)

28.10 M. Mohri, M. Riley: Integrated context-dependent networks in very large vocabulary speech recognition, Proc. 6th Europ. Conf. Speech Commun. Technol. (Eurospeech '99), Budapest (1999)

28.11 S. Ortmanns, H. Ney, A. Eiden: Language-model look-ahead for large vocabulary speech recognition, Proc. Int. Conf. Spoken Lang. Proces. (ICSLP'96), University of Delaware and Alfred I. duPont Institute (1996) pp. 2095–2098

28.12 A. Salomaa, M. Soittola: *Automata-Theoretic Aspects of Formal Power Series* (Springer, New York 1978)

28.13 J. Berstel, C. Reutenauer: *Rational Series and Their Languages* (Springer, Heidelberg 1988)

28.14 W. Kuich, A. Salomaa: *Semirings, Automata, Languages, EATCS Monographs on Theoretical Computer Science*, Vol. 5 (Springer, Berlin, Heidelberg 1986)

28.15 C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, M. Mohri: OpenFST – a General and Efficient Weighted Finite-State Transducer Library, Proceedings of the International Conference on Implementation and Application of Automata, Prague (2007)

28.16 J.E. Hopcroft, J.D. Ullman: *Introduction to Automata Theory, Languages, and Computation* (Addison Wesley, Reading 1979)

28.17 A.V. Aho, R. Sethi, J.D. Ullman: *Compilers, Principles, Techniques and Tools* (Addison Wesley, Reading 1986)

28.18 A.V. Aho, J.E. Hopcroft, J.D. Ullman: *The Design and Analysis of Computer Algorithms* (Addison Wesley, Reading 1974)

28.19 D. Revuz: Minimisation of acyclic deterministic automata in linear time, Theoret. Comput. Sci. **92**, 181–189 (1992)

28.20 M. Mohri: Semiring frameworks and algorithms for shortest-distance problems, J. Automata Lang. Combinat. **7**(3), 321–350 (2002)

28.21 M. Mohri, M. Riley: A weight pushing algorithm for large vocabulary speech recognition, Proc. 7th Europ. Conf. Speech Commun. Technol. (Eurospeech '01), Aalborg (2001)

28.22 D.J. Lehmann: Algebraic structures for transitive closures, Theoret. Comput. Sci. **4**, 59–76 (1977)
28.23 S. Eilenberg: Automata, *Languages and Machines* (Academic, San Diego 1974–1976) vol. A–B
28.24 J. Berstel: *Transductions and Context-Free Languages* (Teubner Studienbucher, Stuttgart 1979)
28.25 C. Allauzen, M. Mohri: Efficient algorithms for testing the twins property, J. Automata Languages Combinat. **8**(2), 117–144 (2003)
28.26 C. Allauzen, M. Mohri: An optimal pre-determinization algorithm for weighted transducers, Theor. Comput. Sci. **328**(1-2), 3–18 (2004)
28.27 M. Mohri: Statistical natural language processing. In *Applied Combinatorics on Words*, ed. by M. Lothaire (Cambridge Univ. Press Cambridge 2005)
28.28 S.M. Katz: Estimation of probabilities from sparse data for the language model component of a speech recogniser, IEEE Trans Acoust. Speech Signal Process. **35**(3), 400–401 (1987)
28.29 C. Allauzen, M. Mohri, B. Roark, M. Riley: A generalized construction of integrated speech recognition transducers, Proc. ICASSP (ICASSP 2004), Montréal (2004)
28.30 M. Riley, F. Pereira, M. Mohri: Transducer composition for context-dependent network expansion, Proc. Eurospeech'97, Rhodes (1997)
28.31 M. Mohri, F. C.N. Pereira: Dynamic compilation of weighted context-free grammars, 36th Annual Meeting ACL and 17th Int. Conf. Computat. Linguist., vol. 2 (1998) p. 891–897
28.32 J.W. Carlyle, A. Paz: Realizations by stochastic finite automaton, J. Comput. Syst. Sci. **5**, 26–40 (1971)
28.33 K. Seymore, R. Rosenfeld: Scalable backoff language models, Proc. ICSLP, Philadelphia (1996)
28.34 S. Kanthak, H. Ney, M. Riley, M. Mohri: A comparison of two LVR search optimization techniques, Proc. Int. Conf. Spoken Lang. Proces. 2002 (ICSLP '02), Denver (2002)
28.35 M. Saraclar, M. Riley, E. Bocchieri, V. Goffin: Towards automatic closed captioning : Low latency real time broadcast news transcription, In Proceedings of the International Conference on Spoken Language Processing (ICSLP'02) (2002)
28.36 A. Ljolje, F. Pereira, M. Riley: Efficient general lattice generation and rescoring, Proc. Europ. Conf. Speech Commun. Technol. (Eurospeech '99), Budapest (1999)
28.37 C. Leggetter, P. Woodland: Maximum likelihood linear regession for speaker adaptation of continuous density HMMs, Comput. Speech Lang. **9**(2), 171–186 (1995)
28.38 M. Riley, W. Byrne, M. Finke, S. Khudanpur, A. Ljolje, J. McDonough, H. Nock, M. Saraclar, C. Wooters, G. Zavaliagkos: Stochastic pronunciation modelling form hand-labelled phonetic corpora, Speech Commun. **29**, 209–224 (1999)
28.39 R. Sproat: Multilingual text analysis for text-to-speech synthesis, J. Nat. Lang. Eng. **2**(4), 369–380 (1997)
28.40 C. Allauzen, M. Mohri, M. Riley: Statistical modeling for unit selection in speech synthesis, In 42nd Meeting of the Association for Computational Linguistics (ACL 2004), Proceedings of the Conference, Barcelona (2004)
28.41 R. M. Kaplan, M. Kay: Regular models of phonological rule systems, Computational Linguistics 20(3) (1994)
28.42 L. Karttunen: The replace operator, In 33rd Meeting of the Association for Computational Linguistics (ACL 95), Proceedings of the Conference, MIT, Cambridge ACL (1995)
28.43 M. Crochemore, W. Rytter: *Text Algorithms* (Oxford Univ. Press, Oxford 1994)
28.44 K.I.I. Culik, J. Kari: Digital images and formal languages. In: *Handbook of Formal Languages*, ed. by G. Rozenberg, A Salomaa (Springer, Berlin, Heidelberg 1997) pp. 599–616

35. Spoken Dialogue Systems

V. Zue, S. Seneff

Spoken dialogue systems are a new breed of interfaces that enable humans to communicate with machines naturally and efficiently using a conversational paradigm. Such a system makes use of many human language technology (HLT) components, including speech recognition and synthesis, natural language understanding and generation, discourse modeling, and dialogue management. In this contribution, we introduce the nature of these interfaces, describe the underlying HLTs on which they are based, and discuss some of the development issues. After providing a historical perspective, we outline some new research directions.

Speech is the most natural means for humans to communicate; nearly all of us can talk and listen to one another without special training. Speech is flexible; it can free our eyes and hands to attend to other tasks. Speech is also very efficient; one can typically speak several times faster than one can type or write. Nowadays, with the pervasiveness of telephones and cell phones, speech is also one of the most inexpensive ways for us to communicate. It is therefore not surprising that speech-based interfaces are in the minds of every techno-visionary, science fiction writer, and Hollywood producer.

Other authors of this volume have discussed automatic speech recognition, an important and as yet unsolved problem in its own right, with a clear set of applications that include document preparation and audio indexing. However, speech recognition by itself is only a part of the interface solution. Human communication is predicated on the assumption that we *understand* one another. In fact, this understanding is often couched in other parts of the dialogue – how can one possibly make sense out of a sentence *what about that one?* without knowledge of the previous dialogue? Many applications that lend themselves to spoken input/output – inquiring about weather or making travel arrangements – are in fact exercises in information access and/or interactive problem solving. The solution is often built up incrementally, with both the user and the computer playing active roles in the *dialogue*. Therefore, several language-based input and output technologies must be developed and integrated to reach this goal. The resulting spoken dialogue systems are the subject of this contribution.

Many speech-based interfaces can be considered conversational, and they may be differentiated by the degree to which the system maintains an active role in the conversation. At one extreme are *system-initiative*, or directed-dialogue transactions where the computer takes complete control of the interaction by requiring that the user answer a set of prescribed questions, much like the touch-tone implementation of interactive voice response (IVR) systems. In the case of air travel planning, for example, a directed-dialogue system could ask the user to *Please say just the departure city*. Since the user's options are severely restricted,

Table 35.1 Transcript of an actual conversation between an agent (A) and a client (C) over the phone. Typical conversational phenomena are annotated on the right

C:	Yeah, [umm] I'm looking for the Buford Cinema.	Disfluency
A:	OK, and you're wanting to know what's showing there or . . .	Interruption
C:	Yes, please.	Confirmation
A:	Are you looking for a particular movie?	
C:	[umm] What's showing.	Clarification
A:	OK, one moment.	Back channel
	. . .	
A:	They're showing *A Troll In Central Park*.	
C:	No.	Inference
A:	Frankenstein.	Ellipsis
C:	What time is that on?	Co-reference
A:	Seven twenty and nine fifty.	
C:	OK, any others?	Fragment
	. . .	

Table 35.2 Transcript of an actual phone conversation between a spoken dialogue system (S) and a novice user (U), using the system for the second time. The system's capabilities are illustrated on the right

U:	I'd like to fly from Boston to Minneapolis on Tuesday – no, Wednesday November 21.	Understand mid-stream correction
S:	Okay, flights from Boston to Minneapolis Saint Paul on Wednesday November twenty first. Can you provide the approximate departure time or airline preference?	Solicit necessary information from user
U:	Yeah, I'd like to fly United and I'd like to leave in the afternoon.	Permit multiple utterances
S:	Okay, United flights from Boston to Minneapolis Saint Paul departing between twelve p.m. and six p.m. on Wednesday November twenty first. One moment please... There are no nonstop flights. There are more than three flights. The earliest departure leaves at 1:45 p.m. and arrives at 6:07 p.m.	Confirm understanding; summarize alternatives
U:	Do you have something leaving around 4:45?	Allow negotiation
S:	Okay, United flights from Boston to Minneapolis Saint Paul leaving at 4:45 p.m. on Wednesday November twenty first. One moment please... What about a flight connecting in Chicago o'Hare leaving at 4:45 p.m., would that work? I also have a flight connecting in Chicago o'Hare leaving at 5:45 p.m.	Articulate pertinent information
U:	4:45.	Understand fragments in context
S:	United flight 1661 connecting to United flight 1107 in Chicago O'Hare has been added to your itinerary. You depart Boston at 4:45 p.m. and arrive in Minneapolis Saint Paul at 9:08 p.m. What date will you be returning on?	
U:	I'd like to return the following Tuesday.	Understand relative dates
	. . .	

successful completion of such transactions is easier to attain, and indeed some successful demonstrations and deployment of such systems have been made [35.1]. However, system-initiated dialogue may cause users to be annoyed due to its inherent inflexibility when the task becomes more complex, involving many turns or exchanges. In fact, user-experience studies [35.2] show that people much preferred *more-natural* user interfaces over system-initiated ones.

At the other extreme are *user-initiative* systems whereby the user has complete freedom in what they say to the system, (e.g., *I want to visit my grandmother*) while the system remains relatively passive, asking only for clarification when necessary. In this case, the user may feel uncertain as to what capabilities exist, and may, as a result, stray quite far from the domain of competence of the system, leading to great frustration because nothing is understood. Lying between these two extremes are systems that incorporate a *mixed-initiative*, goal-oriented dialogue, in which both the user and the computer participate actively to solve a problem interactively using a conversational paradigm. It is this latter mode of interaction that is the primary focus of this contribution.

Of course there is not a black-and-white distinction between mixed- and system-initiative systems. Most system-initiative systems support powerful metacommands such as *back up* and *start over*, which are certainly examples of user initiative. Probably the most effective systems will turn out to be those that can gracefully transition between a more-flexible mode and a mode that guides and restricts the user, based on monitoring user behavior. For instance, when Carnegie Mellon University's *Let's Go!* system [35.3, 4] was deployed to the public, it was changed considerably from its original configuration to increase the amount of user prompting, in order to enhance the yield on successful interactions. It is likely that extensive dialogue context-dependent *help* mechanisms, which would take the form of suggestions or prompts, would be beneficial for transitioning novice users into experts.

One way for developers to discover what form natural human–computer interaction might take is to examine human–*human* interactions during joint problem solving [35.5]. Table 35.1 shows the transcript of a conversation between an agent (A) and a client (C) over the phone. As illustrated by this example, spontaneous dialogue is replete with disfluencies, interruption, confirmation, clarification, ellipsis, co-reference, and sentence fragments. Some of the utterances cannot be understood properly without knowing the context in which they appear. While many of the conversational phenomena illustrated here are still ongoing research challenges, some of the mixed-initiative spoken dialogue systems developed over the past decade can help users solve problems interactively with good success. Table 35.2 shows the transcript of an actual conversation between such a system [35.6] and a novice user calling the system for the second time. This example illustrates the state of the art at the writing of this contribution.

The last decade has witnessed the emergence of some spoken dialogue systems with limited capabilities. Despite this moderate success, the ultimate deployment of such interfaces will require continuing improvement of the core human language technologies (HLTs) and the exploration of many uncharted research territories. In this contribution, we introduce the nature of the spoken dialogue systems, describe the underlying HLTs on which they are based, and discuss some of the development issues. After providing a historical perspective, we outline some of the new research directions. Given space limitations, it is not possible to provide adequate coverage of the entire field. Instead, we will draw from our own experience in developing such systems at MIT since the late 1980s to illustrate our points [35.7–14], and we will provide extensive references. Interested readers are referred to the recent proceedings of the Eurospeech Conference, the International Conference of Spoken Language Processing, the International Conference of Acoustics, Speech, and Signal Processing, the International Symposium on Spoken Dialogue, and other relevant publications [35.15].

35.1 Technology Components and System Development

35.1.1 System Architecture

Figure 35.1 shows the major components of a typical spoken dialogue system. The spoken input is first processed through the speech recognition component. The natural language component, working in concert with the recognizer, produces a meaning representation for the utterance. For the information retrieval application illustrated in this figure, the meaning representation can be used to retrieve the appropriate information in the form of text, tables, and graphics. If the information in the utterance is insufficient or ambiguous, the system may choose to query the user for clarification. If verbal conveyance of the information is desired, then natural language generation and text-to-speech synthesis are utilized to produce the spoken responses. Throughout the process, discourse informa-

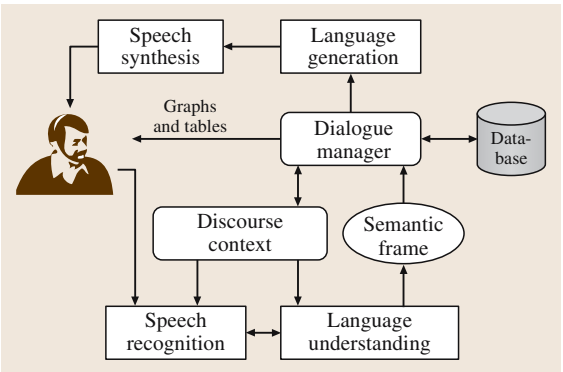


Fig. 35.1 A generic block diagram for a typical spoken dialogue system

tion is maintained and fed back to the speech recognition and language understanding components, so that sentences can be properly understood in context. Finally, a dialogue component manages the interaction between the user and the computer. The nature of the dialogue can vary significantly depending on whether the system is creating or clarifying a query prior to accessing an information database, or perhaps negotiating with the user in a post-information-retrieval phase to relax or somehow modify some aspects of the initial query.

While speech may be the modality of choice, as is the case with phone-based interactions and hands-busy/eyes-busy settings, there are clearly cases where speech is not a good modality, especially, for example, on the output side when the information contains maps, images, or large tables of information that cannot be easily explained verbally. Human communication is inherently multimodal, employing facial, gestural, and other cues to communicate the underlying linguistic message. Thus, speech interfaces should be complemented by visual and sensory motor channels.

Communication can be enhanced by enabling the user to choose among many different modalities, including gesturing, pointing, writing, and typing on the input side [35.16, 17], along with graphics and a talking head on the output side [35.18], to achieve the task in hand in the most natural and efficient manner.

The development of spoken dialogue systems offers a set of new challenges to speech and natural language researchers, and raises several important research issues, some of which will be discussed next.

35.1.2 Spoken Input Processing

Spoken language understanding involves the transformation of the speech signal into a meaning representation that can be used to interact with the specific application back-end. This is typically accomplished in two steps, the conversion of the signal to a set of words (i. e., speech recognition), and the derivation of the meaning from the word hypotheses (i. e., language understanding). A discourse component is often used to properly interpret the meaning of an utterance in the larger context of the interaction.

Often this meaning representation takes the form of a simple list of (attribute: value) pairs encoding user-specified constraints. Thus, the sentence, *Show me flights from San Francisco to Denver leaving in the morning on March ninth* could be represented by the following simple structure:

```
{  action: display
   topic: flights
   source: SFO
   destination: BOS
   departure_interval: morning
   date: Mar09  }
```

Automatic Speech Recognition

Techniques for automatic speech recognition are similar to those described in other contributions in this volume by Young, Mohri et al., and Picheny and Nahamoo. However, some issues that are particular to spoken dialogue input offer additional challenges. Input to spoken dialogue systems is often generated extemporaneously – especially from novice users of these systems. Such spontaneous speech typically contains disfluencies (i. e., unfilled and filled pauses such as *umm* and *aah*, as well as word fragments). In addition, the input utterances are likely to contain words outside the system's working vocabulary – a consequence of the fact that present-day technology can only support the development of systems within constrained domains. Thus far, some attempts have been made to deal with the problem of disfluency. For example, researchers have improved their systems' recognition performance by introducing explicit acoustic models for the filled pauses [35.19, 20]. Similarly, *trash* models have been used to detect the presence of word fragments or unknown words [35.21], and procedures have been devised to learn the new words once they have been detected [35.22, 23]. Suffice it to say, however, that the detection and learning of unknown words continues to be a problem that needs our collective attention. A related topic is utterance- and word-level rejection, in the presence of either out-of-domain queries or unknown words [35.24, 25]. Many systems incorporate some form of confidence scoring to try to identify problematic inputs [35.1, 26, 27]. The system can then try an alternative strategy to either help the user, or back off to a more-directed dialogue and/or one that requires explicit confirmation.

It is perhaps not surprising that some of the first conversational systems available to the general public were accessible via telephone [35.1, 28, 29], in many cases replacing presently existing interactive voice response (IVR) systems. Telephone quality speech is significantly more difficult to recognize than high-quality recordings, both because of the limited bandwidth and the noise and

distortions introduced in the channel [35.30]. The acoustic condition deteriorates further for cellular telephones, either analog or digital.

Natural Language Understanding

Speech recognition systems typically implement linguistic constraints as a statistical language model (i. e., *n*-gram) that specifies the probability of a word given its predecessors. While these language models have been effective in reducing the search space and improving performance, they do not begin to address the issue of speech understanding. On the other hand, most natural language systems are developed with text input in mind; it is usually assumed that the entire word string is known with certainty. This assumption is clearly false for speech input, where many alternative word hypotheses are competing for the same time span in any sentence hypothesis produced by the recognizer (e.g., *euthanasia* and *youth in Asia*,) and some words may be more reliable than others because of varying signal robustness. Furthermore, spoken language is often agrammatical, containing fragments, disfluencies, and partial words. Language understanding systems designed for text input may have to be modified in fundamental ways to accommodate spoken input.

Natural language analysis has traditionally been predominantly syntax-driven – a complete syntactic analysis is performed which attempts to account for *all* words in an utterance. However, when working with spoken material, researchers quickly came to realize that such an approach [35.31–33] can break down dramatically in the presence of unknown words, novel linguistic constructs, recognition errors, and spontaneous speech events.

Due to these problems, many researchers have turned towards more semantic-driven approaches, at least for spoken language tasks in constrained domains. In such approaches, a meaning representation is derived by *spotting* key words and phrases in the utterance [35.34]. While this approach loses the constraint provided by syntax, and may not be able to interpret complex linguistic constructs adequately, the need to accommodate spontaneous speech input has outweighed these potential shortcomings. In fact, many systems have abandoned the notion of achieving a complete syntactic analysis of every input sentence, favoring a more-robust strategy that can still be used to produce an answer when a full parse is not achieved [35.35–37]. This can be accomplished by identifying parsable phrases and clauses, and providing a separate mechanism for gluing them together to form a complete meaning analysis [35.36]. Ideally, the parser includes a probabilistic framework with a smooth transition to parsing fragments when full linguistic analysis is not achievable. Examples of systems that incorporate such *stochastic* modeling techniques can be found in [35.38, 39].

How should the speech recognition component interact with the natural language component in order to obtain the correct meaning representation? One of the most popular strategies is the so-called *N*-best interface [35.40], in which the recognizer proposes its best *N* complete sentence hypotheses one by one, stopping with the first sentence that is successfully analyzed by the natural language component. In this case, the natural language component acts as a filter on *whole sentence* hypotheses. Alternatively, competing recognition hypotheses can be represented in the form of a stochastic word graph or lattice [35.41], which is significantly more compact than an *N*-best list, thus permitting a deeper search if desired. A pruned lattice can easily represent an *N*-best list of several thousand unique utterances. *Sausages* [35.42] provide an even more compact representation, reducing the memory requirement by as much as 100-fold by essentially collapsing a lattice into a sequence of *N*-best word hypotheses.

In an *N*-best list, many of the candidate sentences may differ minimally in regions where the acoustic information is not very robust. While confusions such as *an* and *and* are acoustically reasonable, one of them can often be eliminated on linguistic grounds. In fact, many of the top *N* sentence hypotheses might be eliminated before reaching the end if syntactic and semantic analyses take place early on in the search. One possible solution, therefore, is for the speech recognition and natural language components to be tightly coupled, so that only the acoustically promising hypotheses that are linguistically meaningful are advanced. For example, partial theories can be arranged on a stack, prioritized by score. The most promising partial theories are extended using the natural language component as a predictor of all possible next-word candidates; none of the other word hypotheses are allowed to proceed. Therefore, any theory that completes is guaranteed to parse. Researchers are beginning to find that such a tightly coupled integration strategy can achieve higher performance than an *N*-best interface, often with a considerably smaller stack size [35.43–46]. The future is likely to see increasing use of linguistic analysis at earlier stages in the recognition process.

Discourse Modeling

Human verbal communication is a two-way process involving multiple, active participants. Mutual under-

standing is achieved through direct and indirect speech acts, turn taking, clarification, and pragmatic considerations. Discourse analysis allows a conversational system to understand an utterance in the context of the previous interaction. As such, discourse can be considered to be part of the input processing stage. To communicate naturally, a system must be able to handle phenomena such as deictic (e.g., verbal pointing as in *I'll take the second one*) and anaphoric reference (e.g., using pronouns as in *what's their phone number*) to allow users to refer to items currently in focus efficiently. An effective system should also be able to handle ellipsis and fragments so that a user does not have to fully specify each query. For instance, if a user says, *I want to go from Boston to Denver*, followed with, *show me only United flights*, he/she clearly does not want to see *all* United flights, but rather just the ones that fly from Boston to Denver. The ability to inherit information from preceding utterances is particularly helpful in the face of recognition errors. The user may have asked a complex question involving several restrictions, and the recognizer may have misunderstood a single word, such as a flight number or an arrival time. If a good context model exists, the user can then utter a short correction phrase, and the system will be able to replace just the misunderstood word, preventing the user from having to repeat the entire utterance, running the risk of further errors.

There is a related question of how to *forget* context appropriately. For example, if a user provides a new airline, the remainder of the context can be sensibly preserved. However, if the user provides a new *departure city* not served by the specified airline, pragmatics must play a role in recognizing that the airline in question should be deleted from context.

35.1.3 Spoken Output Processing

On the output side, a spoken dialogue system must be able to convey the information to the user in natural sounding sentences. This task is typically accomplished in two steps. First, the information – whether it be expressed in tabular, database query language, or hypertext forms – must be converted into well-formed natural sentences. If a spoken response is desired, the sentences must then be fed to a text-to-speech (TTS) system to generate the speech signal for playback.

Natural Language Generation

Spoken language generation serves two important roles. First and foremost, it provides a verbal response to the user's queries, which is essential in applications where visual displays are unavailable. In addition, it can provide feedback to the user in the form of a paraphrase, confirming the system's proper understanding of the input query. Although there has been much research on natural language generation (NLG), dealing with the creation of coherent paragraphs [35.47, 48], the language generation component of a spoken dialogue system typically produces the response one sentence at a time, without paragraph-level planning.

Research in language generation for spoken dialogue systems has not received nearly as much attention as has language understanding, especially in the US, perhaps due to the funding priorities set forth by the major government sponsors. In many cases, output sentences are simply word strings, in text or prerecorded acoustic format, that are invoked when appropriate. In some cases, sentences are generated by concatenating templates after filling slots by applying recursive rules along with appropriate constraints [35.49]. There has also been some work using more-corpus-based methods for language generation in order to provide more variation in the surface realization of the utterance [35.50].

Spoken dialogue systems can behave quite differently depending on what input and output modalities are available to the user. In displayless environments such as the telephone, it might be necessary to tailor the dialogue so as not to overwhelm the user with information. When displays are available however, it may be more desirable simply to summarize the information to the user, and to show them a table or image, etc. [35.51]. Similarly, the nature of the interaction will change if alternative input modalities, such as pen or gesture, are available to the user.

Speech Synthesis

The conversion of text to speech is the final stage of output generation (see Part D of this Handbook). TTS systems in the past were primarily rule driven, requiring the system developers to possess extensive acoustic–phonetic and other linguistic knowledge [35.52]. These systems are typically highly intelligible, but suffer greatly in naturalness. In recent years, we have seen the emergence of a new, concatenative approach, brought on by inexpensive computation/storage, and the availability of large corpora [35.53–55]. In this corpus-based approach, units excised from recorded speech are concatenated to form an utterance. The selection of the units is based on a search procedure subject to a predefined distortion measure. The output of these TTS systems is often judged to be more natural than that of rule-based systems [35.56].

The language generation and text-to-speech components of most spoken dialogue systems are not closely coupled; the same text is generated whether it is to be read or spoken. Furthermore, systems typically expect the language generation component to produce a textual surface form of a sentence (throwing away valuable linguistic and prosodic knowledge) and then require the text-to-speech component to produce linguistic analysis anew. In *concept-to-speech* generation [35.57], one *already knows the parse* [35.58], and this knowledge can simplify the synthesizer's task. Such a close coupling can also potentially produce higher-quality output speech than could be achieved with a decoupled system, since it permits finer control of prosody. Whether language generation and speech synthesis components should be tightly integrated, or can remain modular but effectively coupled by augmenting text output with a markup language (e.g., SABLE [35.59]) remains to be seen. Clearly however, these two components would benefit from a shared knowledge base.

35.1.4 Dialogue Management

The dialogue management component of a spoken dialogue system manages the interaction between the user and the computer. The technology for building this component is one of the least developed in the HLT repertoire, especially for the mixed-initiative dialogue systems considered in this contribution. Although there has been some theoretical work on the structure of human–human dialogue [35.60], this has not led to effective insights for building human–machine interactive systems. As mentioned previously, there is also considerable debate in the speech and language research communities about whether modeling human–machine interactions after human–human dialogues is necessary or appropriate [35.61–63].

Dialogue modeling means different things to different people. For some, it includes the *planning* and *problem solving* aspects of human–computer interactions [35.64]. In the context of this chapter, we define dialogue modeling as the planning, for each turn, of the system's side of the conversation, including verbal, tabular, and graphical response, as well as any clarification requests.

Dialogue modeling and management serves many roles [35.65–68]. In the early stages of the conversation the role of the dialogue manager might be to gather information from the user, possibly clarifying ambiguous input along the way, so that, for example, a complete query can be produced for the application database. The dialogue manager must be able to resolve ambiguities that arise due to recognition error (e.g., *Did you say Boston or Austin?*) or incomplete specification (e.g., *On what day would you like to travel?*).

In later stages of the conversation, after information has been accessed from the database, the dialogue manager might be involved in some negotiation with the user. For example, if there were too many items returned from the database, the system might suggest additional constraints to help narrow down the number of choices. Pragmatically, the system must be able to initiate requests so that the information can be reduced to digestible chunks (e.g., *I found ten flights; do you have a preferred airline or connecting city?*).

In addition to these two fundamental operations, the dialogue manager must also inform and guide the user by suggesting subsequent subgoals (e.g., *Would you like me to price your itinerary?*), offer assistance upon request, help relax constraints or provide plausible alternatives when the requested information is not available (e.g., *I don't have sunrise information for Oakland, but in San Francisco ...*), and initiate clarification subdialogues for confirmation. In general the overall goal of the dialogue manager is to take an active role in directing the conversation towards a successful conclusion for the user.

The dialogue manager can influence other system components by, for example, dynamically making dialogue context-dependent adjustments to language models or discourse history. At the highest level, it can help detect the appropriate broad subdomain (e.g., weather, air travel, or urban navigation). Within a particular domain, certain queries could introduce a focus of attention on a subset of the lexicon. For instance, in a dialogue about a trip to France, the initial user utterance, *I'm planning a trip to France*, would allow the system to greatly enhance the probabilities on all the French destinations. Finally, whenever the system asks a directed question, the language model probabilities can be altered so as to favor appropriate responses to the question. For example, when the system asks the user to provide a date of travel, the system could temporarily enhance the probabilities of date expressions in the response.

A challenging area of research is recovery from the inevitable misunderstandings that a system will make [35.69]. Errors could be due to many different phenomena (e.g., acoustics, speaking style, disfluencies, out-of-vocabulary words, parse coverage, or understanding gaps), and it can be difficult to detect that there is a problem, determine what the problem is caused by,

and convey to the user an appropriate response that will fix the problem. This is another responsibility of dialogue management. One of the most challenging aspects of spoken dialogue system development is the design of an appropriate framework for managing the dialogue interaction. For system-initiative designs, it is usually possible to define the dialogue flow in terms of a finite interconnected graph encoding a dialogue state sequence, often referred to as a call-flow diagram. Each state encodes an action to be taken and conditions under which to move to a set of possible next states. However, developers are realizing that even for such scripted systems an explicit enumeration of all possible states and state transitions can quickly become unwieldy for systems with any degree of complexity. Modest extensions to this basic design to support augmented transition networks (ATNs) can lead to a great deal more sharing of common subdialogues among multiple situations, somewhat analogous to subroutine calls in programming languages [35.70].

To handle flexible dialogue interaction, the most straightforward technique is a so-called *electronic-form* (e-form)-based dialogue management strategy. The e-form specifies a set of slots to be filled in, and an agenda specifies the order in which to prompt the user for any missing values [35.71]. An important step towards further generalization can be achieved by decomposing the problem into a static declarative table describing the domain knowledge and a temporally ordered script controlling the turn-by-turn operations [35.72, 73]. The declarative table specifies the entities, attributes, and goals of the domain, while the script executes a sequence of actions based on a specification of Boolean or arithmetic conditions. The conditions test variables in a dialogue state table, which are updated at each turn based on the user query and the system reply. Research at Carnegie Mellon on dialogue management has similarly migrated from the notion of a script-based strategy to an agenda-based strategy, the basic distinction being that the agenda specifies a goal plan in terms of a hierarchy without the explicit temporal order inherent in a script, thus allowing greater flexibility [35.74, 75].

Another very powerful approach to dialogue management, so-called plan-based systems [35.76], is emerging out of the artificial intelligence (AI) community. An AI-style approach to dialogue management also characterizes the information state update (ISU) strategy described in [35.25]. Context information is collected into a central data structure containing, principally, a *dialogue move tree* capturing conversational threads and an *activity tree* representing the past and planned activities. It will be interesting to see if these systems can converge in design with the computer-science-based approaches adopted in the speech community.

35.2 Development Issues

Spoken dialogue systems require first and foremost the availability of high-performance human language technology components such as speech recognition and language understanding. However, the development of these systems also demands that we pay close attention to a host of other issues. While many of these issues may have little to do with human language technologies per se, they are nonetheless crucial to successful system development. In this section, we will outline two of these development issues.

35.2.1 Data Collection

Developing spoken dialogue systems is a classic chicken-and-egg problem. In order to develop the system capabilities, one needs to have a large corpus of data for system development, training, and evaluation. In order to collect data that reflect actual usage, one needs to have a system that users can speak to. Such a corpus is critical for techniques that exploit machine learning, but even when the grammar is manually developed, it is impossible for linguists to imagine the rich variety of user utterances that occur in natural usage.

Figure 35.2 illustrates a typical cycle of system development. For a new domain or language, one must first develop some limited natural language capabilities, thus enabling an *experimenter-in-the-loop*, or *wizard-of-oz*, data collection paradigm, in which an experimenter types the spoken sentences to the system, after removing spontaneous speech artifacts. This process has the advantage of eliminating potential recognition errors. The resulting data are then used for the development and training of the speech recognition and natural language components. As these components begin to mature, it becomes feasible to collect more data using the *system-in-the-loop,* or *wizardless*, paradigm, which is both more realistic and more cost effective. Performance evalua-

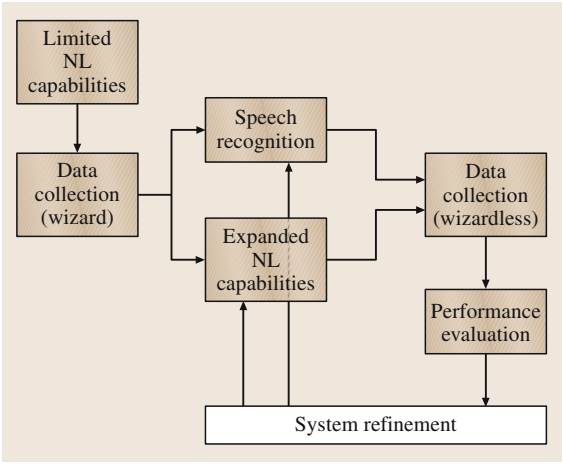


Fig. 35.2 Illustration of data collection procedures

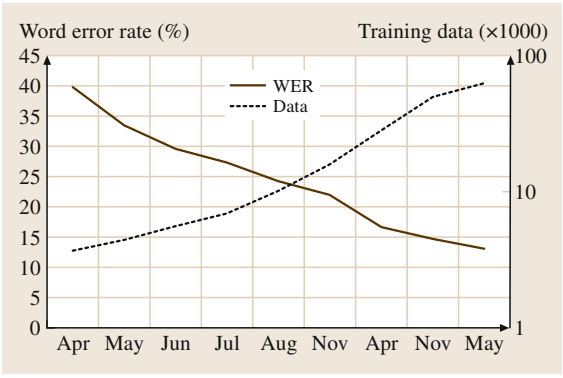


Fig. 35.3 Comparison of recognition performance and the number of utterances collected from real users over time in the MIT weather domain. Note that the x-axis has a non-linear time scale, reflecting the time when new versions of the recognizer were released. WER = word error rate.

tion using newly collected data will facilitate system refinement.

The means and scale of data collection for system development and evaluation has evolved considerably over the last decade. This is true for both the speech recognition and speech understanding communities, and can be seen in many of the systems in the recent automatic railway information systems for Europe (ARISE) project [35.77], and elsewhere. At MIT, for example, the VOYAGER urban navigation system was developed in 1989 by recruiting 100 subjects to come to our laboratory and ask a series of questions to an initial wizard-based system [35.7]. In contrast, the data collection procedure for the more-recent JUPITER weather information system consists of deploying a publicly available system, and recording the interactions [35.14]. There are large differences in the number of queries, the number of users, and the range of issues which the data provide. By using a system-in-the-loop form of data collection, system development and evaluation become iterative procedures. If unsupervised methods were used to augment the system ASR and NLU capabilities, system development could become continuous [35.78].

Figure 35.3 shows, over a two-year period, the cumulative amount of data collected from real users using the MIT JUPITER system and the corresponding word error rates (WER) of our recognizer. Before we made the system accessible through a toll-free number, the WER was about 10% for laboratory-collected data. The WER more than tripled during the first week of data collection. As more data were collected, we were able to build better lexical, language, and acoustic models. As a result, the WER continued to decrease over time. This negative correlation suggests that making the system available to real users is a crucial aspect of system development. If the system can provide real and useful information to users, they will continue to call, thus providing us with a constant supply of useful data. However, in order to get users to actually use the system, it needs to be providing *real* information to the user. Otherwise, there is little incentive for people to use the system other than to play around with it, or to solve toy problem scenarios that may or may not reflect problems of interest to real users.

35.2.2 Evaluation

One of the issues that faces developers of spoken dialogue systems is how to evaluate progress, in order to determine if they have created a usable system. Developers must decide what metrics to use to evaluate their systems to ensure that progress is being made. Metrics can include component evaluations, but should also assess the overall performance of their system.

For systems that conduct a transaction, it is possible to tell whether or not a user has completed a task. In these cases, it is also possible to measure accompanying statistics such as the length of time to complete the task, the number of turns, etc. It has been noted however, that such statistics may not be as important as user satisfaction [35.79, 80]. For example, a spoken dialogue interface may take longer than some alternative, yet users may prefer it due to other factors (less stressful, hands free, etc.). A better form of evaluation might be a measure of whether users liked the system, whether

they called to perform a real task (rather than browsing), and whether they would use it again, or recommend it to others. Evaluation frameworks such as [35.81] attempt to correlate system measurements with user satisfaction, in order to better quantify these effects [35.82].

Although there have been some recent efforts in evaluating language output technologies (e.g., TTS comparisons [35.56]), evaluation for ASR and NLU have been more common since they are more amenable to automatic methods where it is necessary to decide what a correct answer is. ASR evaluation has tended to be the most straightforward, although there are a range of phenomena that are not necessarily obvious how to evaluate (e.g., crosstalk, mumbling, partial words). NLU evaluation can also be performed by comparing some form of meaning representation with a reference. The problem with NLU understanding is that there is no common meaning representation among different research sites, so cross-site comparison becomes difficult. In the DARPA (Defense Advanced Research Projects Agency) spoken language systems (SLS) program for example, the participants ultimately could agree only on comparing to an answer coming from a common database. Unfortunately, this necessarily led to the creation of a large document defining principals of interpretation for all conceivable queries [35.83]. In order to keep the response across systems consistent, systems were restricted from taking the initiative, which is a major constraint on dialogue research.

One way to show progress for a particular system is to perform longitudinal evaluations for recognition and understanding. In the case of JUPITER, as shown in Fig. 35.3, we continually evaluate on standard test sets, which we can redefine periodically in order to keep from tuning to a particular data set [35.14, 84]. Since data continually arrive, it is not difficult to create new sets and re-evaluate older system releases on these new data.

Some systems make use of dialogue context to provide constraints for recognition, for example, favoring candidate hypotheses that mention a date after the system has just asked for a date. Thus, any reprocessing of utterances in order to assess improvements in recognition or understanding performance at a later time need to be able to take advantage of the *same* dialogue context as was present in the original dialogue with the user. To do this, the dialogue context must be recorded at the time of data collection, and reutilized in the subsequent offline processing, in order to avoid giving the original system an unwarranted advantage.

35.3 Historical Perspectives

35.3.1 Large-Scale Government Programs

Research and development of spoken dialogue systems flourished in the 1990s, largely due to government funded research programs in the US and Europe. In the late 1980s the DARPA spoken language systems (SLS) program was initiated in the US [35.83,85], while the Esprit speech understanding and dialog (SUNDIAL) program was underway in Europe [35.86]. The task domains for these two programs were remarkably similar in that both involved database access for travel planning, with the US program concentrating on air travel, and the European program on both air and train travel. All of the systems focused within a narrowly defined area of expertise, and vocabulary sizes were generally limited to several thousand words. By the end of the millennium, these systems could typically run in real time on standard workstations with no additional hardware.

The DARPA SLS program focused entirely on the input side – starting with spoken input and ending with the extraction of the entities from a flight database. By requiring that all system developers use the same database, it has been possible to compare the performance of various systems based on their ability to extract the correct information from the database, using a set of prescribed training and test data, and a set of interpretation guidelines. Unlike the SLS program, however, the European SUNDIAL project aimed at building systems that could be publicly deployed. For this reason, the SUNDIAL project designated dialogue modeling and spoken language generation as integral parts of the research program. As a result, this has led to some interesting advances in Europe in dialogue control mechanisms.

Since the end of the SUNDIAL and SLS programs in 1993, and 1995 respectively, there have been other sponsored programs in spoken dialogue systems. In the European automatic railway information systems for Europe (ARISE) project, participants developed train timetable information systems covering three different languages [35.77]. In the US, a DARPA communicator program [35.87, 88] emphasized dialogue-based interactions incorporating both

speech input and output technologies. One of the properties of this program was that participants used a common system architecture to encourage component sharing across sites [35.89]. Participants in this program developed both their own dialogue domain, and a common complex travel task [35.90].

35.3.2 Some Example Systems

In addition to the research sponsored by these larger programs, there have been many other independent initiatives as well. Some of these systems are research prototypes [35.64, 91, 92], whereas others are actual deployed systems [35.1, 29, 38, 93]. Please see *Zue* and *Glass* [35.94] for a more comprehensive, comparative description of some of them. In the remainder of this section, we will briefly describe a selected subset of the systems to allow readers to gain a sense of the breadth of systems being developed.

There is a strong predominance in the research community on systems supporting database access. Even within this space, the subtopics of air and train travel have dominated [35.6, 64, 93, 95–100], partially as a consequence of the funding opportunities, but also due to the fact that the level of complexity is well matched to research goals, in terms of supporting mixed-initiative dialogue and some degree of planning. The Waxholm project at KTH (the Royal Institute of Technology) in Sweden [35.92, 101] concerns *boat* traffic in the Stockholm archipelago, a slight departure from the main theme. Carnegie Mellon's *Let's Go!* application, providing information on bus schedules within metropolitan Pittsburgh [35.3, 4], is noteworthy because it has been deployed and made available to the general public. This system has a fairly large vocabulary, since there are nearly 15 000 unique bus stops along nearly 2500 routes. The system is accessible via telephone, and receives calls on a regular basis. A recent evaluation experiment showed a similar precipitous drop in performance once the system migrated from the research lab into the public domain as we observed with our Jupiter system. Despite the fact that the raw word error rate on transcripts was 60%, sentence understanding error rate was only 45%, and 44% of the dialogues were successfully completed.

Another system that illustrates migration from research into practical deployment is AT&T's *How May I Help You?* application [35.38]. This system launched a whole new approach to language understanding, where the goal was simply to map a user's first sentence after the *How may I help you?* prompt into one of several semantic classes, in order to route the call to the correct operator or automated subsystems, which can in turn be machine-initiated dialogue systems. A statistical approach to understanding seemed essential for this task, but it has led to widespread adoption of statistical methods for a broader space of applications.

Two other popular topics for dialogue systems are directory assistance [35.26, 29, 102] and map-based navigation [35.7, 103, 104]. While directory assistance is an easier task in terms of dialogue planning than flight scheduling, it can be associated with a very-large-vocabulary name-recognition problem, leading to research on error recovery subdialogues. City guide domains are typically multimodal and critically depend on map displays to provide visual feedback to the user.

35.4 New Directions

35.4.1 User Simulation

One of the most critical barriers to widespread deployment of spoken dialogue systems is the costly development cycle that is critically dependent upon interaction with human users to expose system weaknesses, guide grammar development, and provide domain-specific language model data. Most problematic is the initial stage when the system inadequacies are so apparent that potential users quickly become discouraged and lose interest. One promising strategy to accelerate system development at the early stage involves a user simulation paradigm in which a computer role plays the user side of a simulated conversation. At each turn, the simulated user interprets each system response and decides, using a simple stochastic model, what to say next. In the case of a scripted dialogue this can be pretty straightforward: when the system asks for a date, provide one. Flexible dialogue design poses a more-challenging task to the simulator, but the result is a more-interesting dialogue with greater potential to reveal inadequacies in the dialogue manager that can then be repaired by developers.

The simulated user response can be represented directly in terms of attribute–value pairs, thus bypassing speech recognition and natural language understanding.

Alternatively, it can be converted into a well-formed sentence, using either a formal language generation system or a template-based approach, associating each set of attribute–value pairs with a set of sentence patterns to choose from. The most complete user simulation model involves speech synthesis, producing a waveform that can then be directly processed through the speech recognizer to introduce recognition errors. Recognition errors can also be simulated stochastically.

One of the earliest attempts at user simulation was the work by *Levin* et al. [35.105, 106] where the simulator was used mostly to tune the parameters of a dialogue manager in the flight domain. Simulation experiments by *Lopez-Gozar* [35.107] allowed them to evaluate two different recognition front-ends and two different user confirmation strategies, within a fast-food domain. *Chung* [35.108] developed a user simulation in a flexible restaurant guide domain and showed how it could pinpoint design errors and allow developers to improve system behavior, for example in offering alternatives when the specified constraints yielded no matches from the database. *Chung* et al. [35.109] extended this work to include automatic generation of language-model data for the recognizer, harvested from simulation runs. *Filisko* and *Seneff* [35.110] exploited user simulation to model both compliant and noncompliant behavior in an error recovery subdialogue, for large-vocabulary city-name acquisition tasks. Recently, user simulation has played a critical role in research in Markov decision processes for dialogue modeling [35.111].

35.4.2 Machine Learning and Dialogue Management

Statistical methods and machine learning techniques have been extremely successful in a number of different technologies related to speech processing, e.g., speech recognition, language understanding, information extraction, and language translation. However, these techniques have been unusually slow to penetrate the dialogue management domain. The main reason is that such techniques depend critically on large corpora of manually annotated data, and, for dialogue systems, this translates into the need for detailed, annotated log files for thousands of user dialogues with a pre-existing system. Another roadblock has been uncertainty in how to formulate a tractable machine learning paradigm for the highly heuristic task of dialogue management.

These barriers are beginning to break down in recent years, however, and the community witnessed its first workshop devoted to *Statistical and Empirical Approaches for Spoken Dialogue Systems* in the summer of 2006 [35.112]. A powerful method for side-stepping the data collection issue is to collect synthetic data from user simulation runs, although one ultimately has to confirm that the results carry over to real users. The beauty of simulation is that the user's intended actions are known, so that no manual annotation is required. And the developer can simulate compliant or noncompliant behavior, known or unknown vocabulary choices, etc., in controlled experiments, generating enormous amounts of training data effortlessly.

Levin et al. [35.113] were pioneers in introducing machine learning techniques combined with user simulation, in experiments where they showed that a reasonable policy could be learned in the flight domain through trial and error. Bayesian reinforcement learning, Markov decision processes (MDPs), and Bayesian belief networks (BBNs) [35.114] have evolved into the more-complex partially observable Markov decision process (POMDP) model [35.115], which is beginning to catch on as a method to replace heuristic rules governing dialogue management decisions. However, it is as yet unclear whether these techniques can scale to realistic domains.

Machine learning techniques can in many cases be put to better use in solving a restricted part of the dialogue management problem rather than attempting to replace an existing dialogue manager completely. For example, user simulation data can be used to create a decision tree [35.116], and subsequently to produce a rule-based system for deciding whether to invoke implicit or explicit confirmation, or to seek a spoken spelling of a potentially unknown word [35.110]. *Bohus* and *Rudnicky* [35.27] have developed a data-driven approach that integrates information across multiple turns to aid in the decision process for implicit versus explicit confirmation. Straightforward machine learning techniques have also been successfully applied to the task of deciding whether to reject a user's utterance due to suspicions of gross recognition error [35.25]. *Johnston* and *Bangalore* [35.117] borrowed techniques from the statistical machine translation community to train a set of *edit rules*, leading to more-robust handling of multimodal inputs. Through an automated intent-mapping algorithm, *Tur* [35.118] has shown how annotated data from one domain can be used to build an initial speech understanding model for a new but related domain.

35.4.3 Portability

Creating a robust, mixed-initiative dialogue system can require a tremendous amount of effort on the part of researchers. In order for this technology to ultimately be successful, the process of porting existing technology to new domains and languages must be made easier.

Over the past few years, many research efforts have been directed towards making it easier for non-experts to create new domains. Systems which modularize their dialogue manager try to take advantage of the fact that a dialogue can often be broken down into a smaller set of subdialogues (e.g., dates, addresses), in order to make it easier to construct dialogue interaction for a new domain [35.1, 119, 120]. Researchers at Oregon Graduate Institute have introduced rapid-development toolkits for creating spoken dialogue systems [35.119], which have been used by students to create their own systems [35.121]. MIT's SpeechBuilder framework [35.122] allows novice developers to configure and compile a dialogue system using an intuitive Web-based interface. At Carnegie Mellon University, researchers are exploring ideas to formalize dialogue systems around the concept of *speech graffiti*, arguing that users would be willing to comply to a simplified sublanguage of English in order to gain the advantage of substantially improved recognition and understanding performance [35.123]. A grammar is automatically derived from an appliance specification script, making it relatively easy for developers to create systems based on the *speech graffiti* language syntax. To date, these approaches have been applied only to directed dialogue strategies, and their utility has not been demonstrated beyond toy systems. Much more research is needed in this area before complex dialogue strategies will be able to generalize to new domains with minimal effort and expertise.

Currently, the development of speech recognition and language understanding technologies has been domain and language specific, requiring a large amount of annotated training data. However, it may be costly, or even impossible, to collect a large amount of training data for certain applications or languages. Therefore, an important research topic is to produce a conversational system in a new domain and language given at most a small amount of domain-specific training data. To achieve this goal, the algorithmic aspects of the system must be cleanly separated from the application-specific aspects. Automatic or semiautomatic methods will greatly benefit the acquisition of the acoustic models, language models, grammars, semantic structures for language understanding, and dialogue models required by a new application. Real deployment of multilingual spoken language technology cannot take place without adequately addressing this issue.

35.4.4 Multimodal, Multidomain, and Multilingual Application Development

In this section, we mention several examples of multimodal, multilingual, and multidomain systems representative of the state of the art in these dimensions. Multimodality is an especially exciting topic recently due to the widespread adoption of Internet telephony, offering the potential for effortless interaction at a Web page via microphone input at the computer. Our own research on a multimodal restaurant-domain application has allowed us to explore a number of design issues related to both multimodal input (e.g., drawing a line along a street on a map while asking, *What restaurants are on this street?*), and multimedia output, interfacing with Google maps to provide richly informative visual feedback, which in turn reduces the importance of verbal summarization [35.11]. To provide an integrated search in multimodal understanding, *Johnston* and *Bangalore* [35.124] developed a novel strategy for tight coupling between speech and mouse clicks via a joint interpretation within a weighted finite-state transducer (FST) framework. Another example of a rather unique multimodal dialogue system is WITAS (Wallenberg Laboratory for Information Technology and Autonomous Systems) [35.125], which allows the user to interact with a simulated robotic helicopter, via speech and mouse clicks on a map. The user can instruct the virtual helicopter to fly to different locations, follow vehicles, and deliver goods. A system supporting E-mail applications, AthosMail, which is both multimodal (speech and touch tone) and multilingual (English, Finnish, and Swedish) is described in [35.126]. This system allows users to navigate a set of email messages by voice to read, mark, and delete selected messages.

A challenge in spoken dialogue systems is the notion of supporting multiple domains under a single access point. This introduces the requirement of navigation among the different domains. In our own research [35.127] we have configured systems that allow users to discuss weather, flights, flight status, traffic, and a city guide in a single phone call, although users must refer to each domain by name to explicitly request a domain switch. A system that supports interaction in Chinese in two domains, weather and stock quotes,

is described in [35.128], which addresses both dialogue portability issues and seamless domain switching. A domain-independent goal-oriented table-driven approach to dialogue management is accomplished by externalizing domain-dependent knowledge in a separate table. Another table determines which domain to invoke to answer each question, turn-by-turn, with high accuracy.

35.5 Concluding Remarks

In this paper, we have attempted to outline some of the important research challenges that must be addressed before spoken language technologies can be put to pervasive use. The timing for the development of human language technology is particularly opportune, since the world is mobilizing to develop the information highway that will be the backbone of future economic growth. Human language technology will play a central role in providing an interface that will drastically change the human–machine communication paradigm from *programming* to *conversation*. It will enable users to access, process, manipulate, and absorb a vast amount of information efficiently. While much work needs to be done, the progress made collectively by the community thus far gives us every reason to be optimistic about fielding such systems, albeit with limited capabilities, in the near future.

References

35.1 E. Barnard, A. Halberstadt, C. Kotelly, M. Phillips: A consistent approach to designing spoken-dialog systems, Proc. ASRU Workshop (ASRU, Keystone 1999)

35.2 S.J. Boyce: Natural spoken dialogue systems for telephony applications, Commun. ACM **43**(9), 29–34 (2000)

35.3 A. Raux, B. Langner, D. Bohus, A. Black, M. Eskenazi: Let's Go public! Taking a spoken dialog system to the real world, Proc. Interspeech (2005) pp. 885–888

35.4 A. Raux, B. Langner, A. Black, M. Eskenazi: Let's Go: Improving spoken dialog systems for the elderly and non-natives, Proc. Interspeech (2003) pp. 753–756

35.5 G. Flammia: Discourse Segmentation of Spoken Dialogue: An Empirical Approach. Ph.D. Thesis (MIT, Cambridge 1998)

35.6 S. Seneff, R. Lau, J. Glass, J. Polifroni: The mercury system for flight browsing and pricing, MIT Spoken Language System Group Annual Progress Report (1999) pp. 23–28

35.7 J. Glass, G. Flammia, D. Goodine, M. Phillips, J. Polifroni, S. Sakai, S. Seneff, V. Zue: Multilingual spoken-language understanding in the MIT voyager system, Speech Commun. **17**, 1–18 (1995)

35.8 R. Lau, G. Flammia, C. Pao, V. Zue: WebGalaxy – Integrating spoken language and hypertext navigation, Proc. Eurospeech (1997) pp. 883–886

35.9 H. Meng, S. Busayapongchai, J. Glass, D. Goddeau, L. Hetherington, E. Hurley, C. Pao, J. Polifroni, S. Seneff, V. Zue: A conversational system in the automobile classifieds domain, Proc. ICSLP (1996) pp. 542–545

35.10 S. Seneff, V. Zue, J. Polifroni, C. Pao, L. Hetherington, D. Goddeau, J. Glass: The preliminary development of a displayless Pegasus system, Proc. ARPA Spoken Language Technology Workshop (1995) pp. 212–217

35.11 A. Gruenstein, S. Seneff, C. Wang: Scalable and portable web-based multimodal dialogue interaction with geographical databases, Proc. Interspeech (2006)

35.12 W. Wang, J. Glass, H. Meng, J. Polifroni, S. Seneff, V. Zue: Yinhe: A Mandarin Chinese version of the galaxy system, Proc. Eurospeech (1997) pp. 351–354

35.13 V. Zue, S. Seneff, J. Polifroni, M. Phillips, C. Pao, D. Goddeau, J. Glass, E. Brill: Pegasus: A spoken language interface for on-line air travel planning, Speech Commun. **15**, 331–340 (1994)

35.14 V. Zue, S. Seneff, J. Glass, J. Polifroni, C. Pao, T. Hazen, L. Hetherington: JUPITER: A telephone-based conversational interface for weather information, IEEE Trans. Speech Audio. Process. **8**(1), 85–96 (2000)

35.15 P. Dalsgaard, L. Larsen, I. Thomsen (Eds.): *Proc. ESCA Tutorial and Research Workshop on Spoken Dialogue Systems: Theory and Application* (1995)

35.16 P. Cohen, M. Johnson, D. McGee, S. Oviatt, J. Clow, I. Smith: The efficiency of multimodal interaction: A case study, Proc. ICSLP (1998) pp. 249–252

35.17 S. Seneff, D. Goddeau, C. Pao, J. Polifroni: Multimodal discourse modelling in a multi-user

multi-domain environment, Proc. ICSLP (1996) pp. 188–191
35.18 D. Massaro: *Perceiving Talking Faces: From Speech Perception to a Behavioral Principle* (MIT Press, Cambridge 1997)
35.19 W. Ward: Modelling non-verbal sounds for speech recognition, Proc. DARPA Workshop on Speech and Natural Language (1989) pp. 47–50
35.20 J. Butzberger, H. Murveit, M. Weintraub: Spontaneous speech effects in large vocabulary speech recognition applications, Proc. ARPA Workshop on Speech and Natural Language (1992) pp. 339–344
35.21 L. Hetherington, V. Zue: New words: Implications for continuous speech recognition, Proc. Eurospeech (1991) pp. 475–931
35.22 A. Asadi, R. Schwartz, J. Makhoul: Automatic modelling for adding new words to a large vocabulary continuous speech recognition system, Proc. ICASSP (1991) pp. 305–308
35.23 G. Chung, S. Seneff, C. Wang: Automatic acquisition of names using speak and spell mode in spoken dialogue systems, Proc. HLT-NAACL (2003) pp. 197–200
35.24 C. Pao, P. Schmid, J. Glass: Confidence scoring for speech understanding systems, Proc. ICSLP (1998) pp. 815–818
35.25 M. Gabsdil, O. Lemon: Combining acoustic and pragmatic features to predict recognition performance in spoken dialogue systems, Proc. ACL (2004)
35.26 A. Kellner, B. Rueber, H. Schramm: Using combined decisions and confidence measures for name recognition in automatic directory assistance systems, Proc. ICSLP (1998) pp. 2859–2862
35.27 D. Bohus, A. Rudnicky: Constructing accurate beliefs in spoken dialog systems, Proc. ASRU (2005) pp. 272–277
35.28 V. Souvignier, A. Kellner, B. Rueber, H. Schramm, F. Seide: The thoughtful elephant: Strategies for spoken dialogue systems, IEEE T. Speech Audi. P. **8**(1), 51–62 (2000)
35.29 R. Billi, R. Canavesio, C. Rullent: Automation of Telecom Italia Directory Assistance Service: Field trial results, Proc. IVTTA (1998) pp. 11–16
35.30 R.J. Lippmann: Speech recognition by humans and machines, Speech Commun. **22**(1), 1–15 (1997)
35.31 R. Bobrow, R. Ingria, D. Stallard: Syntactic and semantic knowledge in the DELPHI unification grammar, Proc. DARPA Speech and Natural Language Workshop (1990) pp. 230–236
35.32 S. Seneff: TINA: A natural language system for spoken language applications, Comput. Linguist. **18**(1), 61–86 (1992)
35.33 J. Dowding, J. Gawron, D. Appelt, J. Bear, L. Cherny, R. Moore, D. Moran: Gemini: A natural language system for spoken language understanding, Proc. ARPA Workshop on Human Language Technology (1993) pp. 21–24
35.34 W. Ward: The CMU air travel information service: Understanding spontaneous speech, Proc. ARPA Workshop on Speech and Natural Language (1990) pp. 127–129
35.35 E. Jackson, D. Appelt, J. Bear, R. Moore, A. Podlozny: A template matcher for robust NL interpretation, Proc. DARPA Speech and Natural Language Workshop (1991) pp. 190–194
35.36 S. Seneff: Robust parsing for spoken language systems, Proc. ICASSP (1992) pp. 189–192
35.37 D. Stallard, R. Bobrow: Fragment processing in the DELPHI system, Proc. DARPA Speech and Natural Language Workshop (1992) pp. 305–310
35.38 A. Gorin, G. Riccardi, J. Wright: How may I help you?, Speech Commun. **23**, 113–127 (1997)
35.39 S. Miller, R. Schwartz, R. Bobrow, R. Ingria: Statistical language processing using hidden understanding models, Proc. ARPA Speech and Natural Language Workshop (1994) pp. 278–282
35.40 Y. Chow, R. Schwartz: The N-Best Algorithm: An efficient procedure for finding top N sentence hypotheses, Proc. ARPA Workshop on Speech and Natural Language (1989) pp. 199–202
35.41 L. Hetherington, M. Phillips, J. Glass, V. Zue: A word network search for continuous speech recognition, Proc. Eurospeech (1993) pp. 1533–1536
35.42 L. Mangu, E. Brill, A. Stolcke: Finding consensus among words: Lattice-based word error minimization, Proc. Eurospeech (1999)
35.43 D. Goodine, S. Seneff, L. Hirschman, M. Phillips: Full integration of speech and language understanding in the MIT spoken language system, Proc. Eurospeech (1991) pp. 845–848
35.44 D. Goddeau: Using probabilistic shift-reduce parsing in speech recognition systems, Proc. ICSLP (1992) pp. 321–324
35.45 W. Ward: Integrating semantic constraints into the SPHINX-II recognition search, Proc. ICASSP (1994) pp. 17–20
35.46 R. Moore, D. Appelt, J. Dowding, J. Gawron, D. Moran: Combining linguistic and statistical knowledge sources in natural-language processing for ATIS, Proc. ARPA Spoken Language Systems Workshop (1995) pp. 261–264
35.47 E. Reiter, R. Dale: *Building Natural Language Generation Systems* (Cambridge Univ. Press, Cambridge 2000)
35.48 D. McDonald, L. Bolc (Eds.): *Natural Language Generation Systems (Symbolic Computation Artificial Intelligence)* (Springer, Heidelberg, Berlin 1998)
35.49 J. Glass, J. Polifroni, S. Seneff: Multilingual language generation across multiple domains, Proc. ICSLP (1994)
35.50 A. Oh: Stochastic Natural Language Generation for Spoken Dialog Systems. M.S. Thesis (CMU, Mt. Pleasant 2000)

35.51 V. Demberg, J. Moore: Information presentation in spoken dialogue systems, Proc. EACL (2006)

35.52 D. Klatt: Review of text-to-speech conversion for English, J. Acoust. Soc. Am. **82**(3), 737–793 (1987)

35.53 Y. Sagisaka, N. Kaiki, N. Iwahashi, K. Mimura: ATR ν -talk speech synthesis system, Proc. ICSLP (1992) pp. 483–486

35.54 M. Beutnagel, A. Conkie, J. Schroeter, Y. Stylianou, A. Syrdal: The AT&T next-gen TTS system, Proc. ASA (ASA, Berlin 1999)

35.55 X. Huang, A. Acero, J. Adcock, H.W. Hon, J. Goldsmith, J. Liu, M. Plumpe: Whistler: A trainable text-to-speech system, Proc. ICSLP (1996) pp. 2387–2390

35.56 J. van Santen, L. Pols, M. Abe, D. Kahn, E. Keller, J. Vonwiller: Report on the 3rd ESCA TTS workshop evaluation procedure, Proc. 3rd ESCA Workshop on Speech Synthesis (1998) pp. 329–332

35.57 K. McKeown, S. Pan, J. Shaw, D. Jordan, B. Allen: Language generation for multimedia healthcare briefings, Proc. Applied Natural Language Proc (1997)

35.58 A. Black, K. Lenzo: Limited domain synthesis, Proc. ICSLP (2000)

35.59 R. Sproat, A. Hunt, M. Ostendorf, P. Taylor, A. Black, K. Lenzo, M. Edgington: Sable: A standard for TTS markup, Proc. ICSLP (1998) pp. 1719–1722

35.60 B. Grosz, C. Sidner: Plans for discourse, Intentions in Communication (MIT Press, 1990)

35.61 D. Thomson, J. Wisowaty: User confusion in natural language services, Proc. ESCA Workshop on Interactive Dialogue in Multi-Modal Systems (1999) pp. 189–196

35.62 D. Sadek: Design considerations on dialogue systems: From theory to technology – The case of Artimis, Proc. ESCA Workshop on Interactive Dialogue in Multi-Modal Systems (1999) pp. 173–188

35.63 L. Boves, E. den Os: Applications of speech technology: Designing for usability, Proc. IEEE Worshop on ASR and Understanding (1999) pp. 353–362

35.64 J. Allen, L. Schubert, G. Ferguson, P. Heeman, C. Hwang, T. Kato, W. Light, N. Martin, B. Miller, M. Poesio, D. Traum: The TRAINS project: A case study in defining a conversational planning agent, J. Exp. Theor. Artif. Intell. **7**, 7–48 (1995)

35.65 M. Denecke, A. Waibel: Dialogue strategies guiding users to their communicative goals, Proc. Eurospeech (1997) pp. 2227–2230

35.66 L. Devillers, H. Bonneau-Maynard: Evaluation of dialog strategies for a tourist information retrieval system, Proc. ICSLP (1998) pp. 1187–1190

35.67 S. Rosset, S. Bennacef, L. Lamel: Design strategies for spoken language dialog systems, Proc. Eurospeech (1999) pp. 1535–1538

35.68 A. Rudnicky, E. Thayer, P. Constantinides, C. Tchou, R. Shern, K. Lenzo, W. Xu, A. Oh: Creating natural dialogs in the Carnegie Mellon communicator system, Proc. Eurospeech (1999) pp. 1531–1534

35.69 E. Filisko, S. Seneff: Error detection and recovery in spoken dialogue systems, Proc. Workshop on Spoken Language Understanding for Conversational Systems and Higher Level Linguistic Knowledge for Speech Processing (2004)

35.70 G. Di Fabbrizio, C. Lewis: Florence: A dialogue manager framework for spoken dialogue systems, Proc. ICSLP (2004)

35.71 D. Goddeau, H. Meng, J. Polifroni, S. Seneff, S. Busayapongchai: A form-based dialogue manager for spoken language applications, Proc. ICSLP (1996) pp. 701–704

35.72 R. Pieraccini, E. Levin, W. Eckert: AMICA: The AT&T mixed initiative conversational architecture, Proc. Eurospeech (1997) pp. 1875–1879

35.73 S. Seneff: Response planning and generation in the MERCURY flight reservation system, Comput. Speech Lang., Vol. 16 (2002) pp. 283–312

35.74 P. Constantinides, S. Hansma, A. Rudnicky: A schema-based approach to dialog control, Proc. ICSLP (1998) pp. 409–412

35.75 X. Wei, A. Rudnicky: Task-based dialog management using an agenda, Proc. ANLP/NAACL Workshop on Conversational Systems (2000)

35.76 J. Allen, D. Byron, M. Dzikovska, G. Ferguson, L. Galescu, A. Stent: An architecture for a generic dialogue shell, Nat. Lang. Eng. **6**(3), 1–16 (2000)

35.77 E. den Os, L. Boves, L. Lamel, P. Baggia: Overview of the ARISE project, Proc. Eurospeech (1999) pp. 1527–1530

35.78 T. Kemp, A. Waibel: Unsupervised training of a speech recognizer: Recent experiments, Proc. Eurospeech (1999) pp. 2725–2728

35.79 A. Rudnicky, M. Sakamoto, J. Polifroni: Evaluating spoken language interaction, Proc. DARPA Speech and Natural Language Workshop (1989) pp. 150–159

35.80 L. Lamel, S. Bennacef, J.L. Gauvain, H. Dartigues, J. Temem: User evaluation of the mask kiosk, Proc. ICSLP (1998) pp. 2875–2878

35.81 M. Walker, D. Litman, C. Kamm, A. Abella: PARADISE: A general framework for evaluating spoken dialogue agents, Proc. ACL/EACL (1997) pp. 271–280

35.82 M. Walker, J. Boland, C. Kamm: The utility of elapsed time as a usability metric for spoken dialogue systems, Proc. ASRU Workshop (ASRU, Keystone 1999) pp. 1167–1170

35.83 L. Hirschman: Multi-site data collection for a spoken language corpus, Proc. DARPA Workshop on Speech and Natural Language (1992) pp. 7–14

35.84 J. Polifroni, S. Seneff, J. Glass, T. Hazen: Evaluation methodology for a telephone-based conversational system, Proc. Int. Conf. on Lang. Resources and Evaluation (1998) pp. 42–50

35.85 D. Pallett, J. Fiscus, W. Fisher, J. Garafolo, B. Lund, A. Martin, M. Pryzbocki: Benchmark tests for the ARPA spoken language program, Proc. ARPA Spo-

ken Language Systems Technology Workshop (1995) pp. 5–36
35.86 J. Peckham: A new generation of spoken dialogue systems: Results and lessons from the SUNDIAL project, Proc. Eurospeech (1993) pp. 33–40
35.87 M. Walker, J. Aberdeen, J. Boland, E. Bratt, J. Garofolo, L. Hirschman, A. Le, S. Lee, S. Narayanan, K. Papineni, B. Pellom, J. Polifroni, A. Potamianos, P. Prabhu, A. Rudnicky, G. Sanders, S. Seneff, D. Stallard, S. Whittaker: DARPA communicator dialog travel planning systems: The June 2000 data collection, Proc. Eurospeech (2001)
35.88 M. Walker, A. Rudnicky, R. Prasad, J. Aberdeen, E.O. Bratt, J. Garofolo, H. Hastie, A. Le, B. Pellom, A. Potamianos, R. Passonneau, S. Roukos, G. Sanders, S. Seneff, D. Stallard: DARPA communicator: Cross-system results for the 2001 evaluation, Proc. ICSLP (2002) pp. 273–276
35.89 S. Seneff, E. Hurley, R. Lau, C. Pao, P. Schmid, V. Zue: GALAXY-II: A reference architecture for conversational system development, Proc. ICSLP (1998) pp. 931–934
35.90 M. Eskenazi, A. Rudnicky, K. Gregory, P. Constantinides, R. Brennan, C. Bennett, J. Allen: Data collection and processing in the Carnegie Mellon communicator, Proc. Eurospeech (1999) pp. 2695–2698
35.91 D. Jurafsky, C. Wooters, G. Tajchman, J. Segal, A. Stolcke, E. Fosler, N. Morgan: The Berkeley restaurant project, Proc. ICSLP (1994) pp. 2139–2142
35.92 M. Blomberg, R. Carlson, K. Elenius, B. Granstrom, J. Gustafson, S. Hunnicutt, R. Lindell, L. Neovius: An experimental dialogue system: Waxholm, Proc. Eurospeech (1993) pp. 1867–1870
35.93 H. Aust, M. Oerder, F. Seide, V. Steinbiss: The Philips automatic train timetable information system, Speech Commun. **17**, 249–262 (1995)
35.94 V. Zue, J. Glass: Conversational interfaces: Advances and challenges, Proc. IEEE, Vol. 88 (2000), Special Issue on Spoken Language Processing
35.95 J. Kowtko, P. Price: Data collection and analysis in the air travel planning domain, Proc. DARPA Speech and Natural Language Workshop (1989)
35.96 G. Castagnieri, P. Baggia, M. Danieli: Field trials of the Italian ARISE train timetable system, Proc. IVTTA (1998) pp. 97–102
35.97 M. Cohen, Z. Rivlin, H. Bratt: Speech recognition in the ATIS domain using multiple knowledge sources, Proc. DARPA Spoken Language Systems Technology Workshop (1995) pp. 257–260
35.98 L. Lamel, S. Rosset, J.L. Gauvain, S. Bennacef, M. Garnier-Rizet, B. Prouts: The LIMSI ARISE system, Proc. IVTTA (1998) pp. 209–214
35.99 A. Sanderman, J. Sturm, E. den Os, L. Boves, A. Cremers: Evaluation of the Dutch train timetable information system developed in the ARISE project, Proc. IVTTA (1998) pp. 91–96
35.100 J. Sturm, E. den Os, L. Boves: Dialogue management in the Dutch ARISE train timetable information system, Proc. Eurospeech (1999) pp. 1419–1422
35.101 R. Carlson, S. Hunnicutt: Generic and domain- specific aspects of the Waxholm NLP and dialogue modules, Proc. ICSLP (1996) pp. 677–680
35.102 B. Buntschuh: VPQ: A spoken language interface to large scale directory information, Proc. ICSLP (1998) pp. 2863–2866
35.103 M. Johnston, S. Bangalore, G. Vasireddy, A. Stent, P. Ehlen, M. Walker, S. Whattaker, P. Maloor: MATCH: An architecture for multimodal dialogue systems, Proc. ACL (2002) pp. 376–383
35.104 S. Oviatt: Multimodal interfaces for dynamic interactive maps, Proc. CHI (1996) pp. 95–102
35.105 E. Levin, R. Pieraccini, W. Eckert: A stochastic model of human–machine interaction for learning dialogue strategies, IEEE T. Speech Audi. P. **8**(1), 11–23 (2000)
35.106 W. Eckert, E. Levin, R. Pieraccini: User modelling for spoken dialogue system evaluation, Proc. ASRU (1997) pp. 80–87
35.107 R. Lopez-Cozar, A. De la Torre, J.C. Segura, A.J. Rubio: Assessment of dialogue systems by means of a new simulation technique, Speech Commun. **40**, 387–407 (2003)
35.108 G. Chung: Developing a flexible spoken dialog system using simulation, Proc. ACL (2004) pp. 63–70
35.109 G. Chung, S. Seneff, C. Wang: Automatic induction of language model data for a spoken dialogue system, Proc. 6th SIGdial Workshop on Discourse and Dialogue Lisbon (2005)
35.110 E. Filisko, S. Seneff: Learning decision models in spoken dialogue systems via user simulation, Proc. AAAI Workshop: Statistical and Empirical Approaches for Spoken Dialog Systems (2006)
35.111 J. Schatzmann, K. Georgila, S. Young: Quantitative evaluation of user simulation techniques for spoken dialogue systems, Proc. Workshop on Discourse and Dialogue (2005)
35.112 P. Poupart, S. Seneff, J. Williams, S. Young: Cochairs. In: *Statistical and Empirical Approaches for Spoken Dialogue Systems*, ed. by AAAI (AAAI, Menlo Park 2006), Tech. Rep. WS-06-14
35.113 E. Levin, R. Pieraccini, W. Eckert: Using Markov decision process for learning dialogue strategies, Proc. ICASSP (1998) pp. 201–204
35.114 H. Meng, C. Wai, R. Pieraccini: The use of belief networks for mixed-initiative dialog modeling, IEEE T. Speech Audi. P. **11**(6), 757–773 (2003)
35.115 J. Williams, S. Young: Scaling POMDPs for dialog management with composite summary point-based value iteration (CSPBVI), Proc. AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems (2006) pp. 7–12

35.116 W. Cohen: Fast effective rule induction, Proc. 12th Int. Conf. Machine Learning (1995) pp. 115–123
35.117 M. Johnston, S. Bangalore: Learning edit machines for robust multimodal understanding, Proc. ICASSP (2006) pp. I617–I620
35.118 G. Tur: Multitask learning for spoken language understanding, Proc. ICASSP (2006) pp. I585–I588
35.119 S. Sutton: Universal speech tools: The CSLU toolkit, Proc. ICSLP (1998) pp. 3221–3224
35.120 J. Glass, S. Seneff: Flexible and personalizable mixed-initiative dialogue systems, HLT-NAACL 2003 Workshop on Research Directions in Dialogue Processing (2003)
35.121 S. Sutton, E. Kaiser, A. Cronk, R. Cole: Bringing spoken language systems to the classroom, Proc. Eurospeech (1997) pp. 709–712
35.122 J. Glass, E. Weinstein, S. Cyphers, J. Polifroni, G. Chung, M. Nakano: A framework for developing conversational user interfaces, Proc. CADUI (2004) pp. 354–365
35.123 T. Harris, R. Rosenfeld: A universal speech interface for appliances, Proc. ICSLP (2004)
35.124 M. Johnston, S. Bangalore: Finite-state multimodal parsing and understanding, Proc. 18th Intrenational Conference on Computational Linguistics (2000) pp. 369–375
35.125 O. Lemon, A. Gruenstein, S. Peters: Collaborative activities and multitasking in dialogue systems, Traitement Automatique des Langues **43**(2), 131–154 (2002)
35.126 M. Turunen, E.-P. Salonen, M. Hartikainen, J. Hakulinen, W. Black, A. Ramsay, A. Funk, A. Conroy, P. Thompson, M. Stairmand, L.K. Jokenen, J. Rissanen, K. Kanto, A. Kerminen, B. Gamback, M. Cheadle, F. Olsson, M. Sahlgren: AthosMail – A – multilingual adaptive spoken dialogue system for e-mail domain, Proc. COLING Workshop (2004)
35.127 S. Seneff, R. Lau, J. Polifroni: Organization, communication, and control in the GALAXY-II conversational system, Proc. Eurospeech (1999) pp. 1271–1274
35.128 W.-T. Hsu, H.-M. Wang, Y.-C. Lin: The design of a multi-domain Chinese dialogue system, Proc. ISCSLP (2002)

34. The Business of Speech Technologies

J. Wilpon, M. E. Gilbert, J. Cohen

With the fast pace of developments of communications networks and devices, immediate and easy access to information and services is now the expected norm. Several critical technologies have entered the marketplace as key enablers to help make this a reality. In particular, speech technologies, such as speech recognition and natural language understanding, have changed the landscape of how services are provided by businesses to consumers forever. In 30 short years, speech has progressed from an idea in research laboratories across the world, to a multibillion-dollar industry of software, hardware, service hosting, and professional services. Speech is now almost ubiquitous in cell phones. Yet, the industry is still very much in its infancy with its focus being on simple *low hanging fruit* applications of the technologies where the current state of technology actually fits a specific market need, such as voice enabling of call center services or voice dialing over a cell phone.

With broadband access to networks (and therefore data), anywhere, anytime, and using any device, almost a reality, speech technologies will continue to be essential for unlocking the potential that such access provides. However, to unlock this potential, advances in basic speech technologies beyond the current state of the art are essential. In this chapter, we review the *business* of speech technologies and its development since the 1980s. How did it start? What were the key inventions that got us where we are, and the services innovations that supported the industry over the past few decades? What are the future trends on how speech technologies will be used? And what are the key technical challenges researchers must address and resolve for the industry to move forward to meet this vision of the future? This chapter is by no means meant to be exhaustive, but it gives the reader an understanding of speech technologies, the speech business, and areas where continued technical invention and innovation will be needed before the ubiquitous use of speech technologies can be seen in the marketplace.

34.1 Introduction

In 1969, the influential John Pierce wrote an article questioning the prospects of speech recognition, criticizing the *mad inventors and unreliable engineers* working in the field. In his article, entitled *Whither speech recognition?*, Pierce argued that speech recognition was futile because the task of speech understanding is too difficult for any machine [34.1]. Such a speech understanding system would require tremendous advances in linguistics, natural language, and knowledge of everyday human experiences. In this prediction he was completely correct. Furthermore, this ultimate goal is still not within sight in 2006. Pierce went on to describe the motivation for speech recognition research: 'The attraction [of speech recognition] is perhaps similar to the attraction of schemes for turning water into gasoline, extracting gold from the sea, curing cancer, or going to the moon.' His influential article was successful in curtailing, but not stopping, speech recognition research.

What Pierce failed to foretell was that even limited success in speech recognition would have interesting and important applications, especially within the telecommunications industry. In the decades since Pierce's article, tens of thousands of speech recognition systems have been put into use, and have proven to be important enabling technologies. From speech research to speech engine products, delivery platforms, application development, application tuning and maintenance, and service hosting – speech technologies are responsible for generating new revenue streams and vast cost savings. Players range from small startups and universities to Fortune 500 companies – each with its own particular focus on the revenue chain created by a group of technologies.

In this chapter, we present a view on the applications and services that are driving the need for speech recognition and natural language understanding technologies with a focus on those opportunities that provide the broadest value to people and businesses. Current and future needs are described and examined in terms of their strengths, limitations, and the degree to which stakeholder needs have been or have yet to be met. We start with brief high-level business and technical overviews. Next we drill down into the specific opportunity spaces in several markets – Sect. 34.1 network-based, Sect. 34.2 mobile devices, Sect. 34.3 government, and Sect. 34.4 new emerging markets including multimodal and multimedia applications. And finally, we present our view on where we believe the speech industry is heading. This chapter is not meant to be exhaustive, but to give the reader an understanding of speech technologies, the speech business, and areas where continued technical invention and innovation will be needed before the ubiquitous use of speech technologies can be seen in the marketplace.

34.1.1 Economic Value of Network-Based Speech Services

America's increasing mobility on and off the job lead users to demand access to automated services anytime and anywhere using any access device. The selection of user interface mode or combination of modes depends on the device, the task, the environment, and the user's abilities or preferences. As such, industry demand has been increasingly pointed toward a wide range of voice-enabled services, such as:

- Consumer communication (voice dialing, unified messaging, voice portals, and access to services for people with disabilities)
- E-commerce (business-to-business and business-to-consumer)
- Call center automation (help lines and customer care)
- Enterprise communication (unified messaging and enterprise sales)

Network-based voice-enabled services imply scale. That is, the ability to automate parts of or entire transactions from billions of calls per year provides the opportunity to radically change the cost structure of companies' telecommunications and call center operations. Speech-enabled automation for network services can vastly enhance user experience while optimizing revenue potential and cost savings for businesses; hence their increased popularity.

Decreased Cost

Robust automation of call center functions radically reduces agent counts and operational overhead, while innovative, scalable technology reduces servicing costs. Current call centers, for example, are characterized by fragmentation of channels, limited customer insight, and high dependency on live agent interactions, resulting in high costs, and inconsistent customer experiences. Dramatic success can be achieved by utilizing speech-based services. For example, call center human agents can cost

5.50–12.00 $ per call , while speech-enabled automation can cost less than 0.20 $ per call and are 40% faster on average.

Network-hosted services reduce barriers to entry by limiting the capital spending and the ongoing maintenance costs for companies. In addition, given the complexity and maturity level of speech technologies and services, it is increasingly more difficult and expensive for individual companies:

1. to be responsible for maintaining up-to-date technologies on their platform, and
2. to be able to design ever more-complex speech solutions to meet their call center needs.

Improved Customer Experience

As customers experience more-effective natural, conversational, and personalized interaction that understands and fulfills their requests, they begin to accept automation. Wait and call times are reduced even during peak periods because of improved correct routing rates, increased call completion rates, and a reduction in the rate of agent transfers. This results in happier customers.

Revenue Generation

Redefined business models exploit new revenue opportunities with personal and customized services. New services such as information search and retrieval services (e.g., access to information available on the Internet), are not viable in a mobile environment without the use of speech technologies. Additionally, as will be discussed below, speech technologies provide the ability to simplify complex interactive voice response (IVR) menu structures. Therefore, businesses can offer an increased number of automated service opportunities (e.g., e-commerce) than could otherwise be achieved using touch-tone interfaces.

34.1.2 Economic Value of Device-Based Speech Applications

Small devices have been a challenging platform for speech systems because of their limited computing capability, short-lived batteries, imperfect audio systems, and small audiences. Starting with the advent of the cell phone is the mid 1990s, all of these constraints are lifting.

The number of cell phone users has exploded worldwide. The cell phone is an essential part of today's communication infrastructure in the developed world, and the cell-phone infrastructure is invading the developing world because of the ease of deployment and the economies of scale due to tremendous volumes. Audio systems are carefully crafted in these devices because of the essential part that voice quality plays in telephony. Battery life has increased both because of technical advances, and due to engineering discipline caused by consumer-led product analysis. Finally, computational abilities are growing rapidly due to the demands for high-bandwidth connectivity, multimodal use of devices, and applications ranging from browsers to music players.

While there are improvements in mobile enterprise systems and personal digital assistants (PDAs), the economic gorilla in this space is, and will remain, the cell phone and associated smart phones and feature phones. (Smart phones have open operating systems, while feature phones appear to have the capabilities of smart phones while generally retaining the proprietary operating systems of their consumer cell-phone counterparts.)

The value of the cell-phone market is immense. Gartner predicts that in 2009, 1 billion cell phones will be sold, at an average price (not necessarily to the consumer) of 174 $. Given that speech services will be available on about half that number (say 500 million), and that second-tier (nonprimary) software services can command 1–2% of the value of the devices in which they are embedded, we estimate the economic value of the speech market for embedded devices to be at least 870 million $ per year.

Note that cell phones are replaceable items with an expected 2.5 year lifespan, so this economic market will continue. Specialized services (games, dictation, or other high-value applications) can add substantially to the value of speech services, so our estimate should be treated as a lower limit. In short, the embedded speech market is about a billion dollars a year.

34.1.3 Technology Overview

Technology Landscape

Progress in speech and language technologies along with advances in computing and electronic devices over the past 50 years have resulted in a spectrum of innovative human/computer applications and services. The field has been transformed from the simple speaker-dependent isolated-digit recognition systems that were demonstrated in 1952 by *Davis*, *Biddulph*, and *Balashek* of Bell Labs [34.2], to today's very large-vocabulary speaker-independent context-dependent, IBM continuous speech recognition and understanding systems as demonstrated by AT&T, BBN,

Cambridge, Carnegie Mellon University (CMU), among others [34.3–7]. This steady progress in acoustic and language modeling, robust feature extraction, search and network optimization has enabled speech-based applications to advance from the simplest command-and-control applications deployed in the 1980s, to those in use today for network services and speech data mining that require upwards of 1 M word vocabularies. A few of the key technology innovations include:

1. Acoustic/phonetic: initially studied by *Davis* et al. [34.2] to extract salient spectral features from filter-bank analysis that can better correlate phoneme sets, such as vowels.
2. Linear predictive coding (LPC): developed by Itakura and Atal in the late 1960s for speech coding, and has been shown to be central for extracting features, such as cepstrum, from speech [34.8].
3. Dynamic programming (DP): proposed by *Vintsyuk* [34.9] for the alignment of two speech utterances of different lengths.
4. Hidden Markov models (HMMs): proven to converge by Baum, Petrie, Soules and Weiss, and later developed by *Jellink* and *Baker* at IBM [34.8,10] and popularized by Larry *Rabiner* at Bell Labs [34.11] for modeling speech – a departure from template-based methods that were previously proposed by Bell Labs. The combination of HMMs and Gaussian mixture models (GMMs) are the heart and soul of most of today's speech recognition systems.
5. Stochastic language models (SLMs): proposed by *Jellink* at IBM [34.12] for language modeling in continuous speech recognition.
6. Discriminative training (DT): methods including maximum mutual information (MMI) and minimum classification error training (MCE) that aim to minimize the expected recognition error rate [34.13].
7. Large-margin classifiers: methods including support vector machines (SVMs) and boosting, which attempt to maximize the margins between correct and incorrect classes, have proved to be essential for spoken language understanding [34.14].
8. Neural networks: developed for speech recognition at Institut Dalle Molle d'Intelligence Artificielle Perceptive (IDIAP), International Computer Science Institute (ICSI) and Oregon Graduate Institute of Science and Technology (OGI) by Bourlard, Morgan, and Hermansky, these nonlinear classifiers proved useful in providing rapid probability estimates for use in HMM decoders, and allowed the use of a posterior probabilities rather than likelihoods in search.
9. Digital signal processing: widely practiced by electrical and communications engineers worldwide, this opened the field of speech for moderate-computation front-end processing using both fast Fourier transform (FFT) and multiscale front-end processors.
10. JAVA, Windows Mobile CE, and BREW: these operating system extensions allowed third-party applications to be added to both standard handsets and smart phones, thereby accelerating innovation in mobile platform offerings.
11. VTLN (vocal-tract-length normalization): a one-parameter transformation which accounts, to first order, for varying vocal-tract lengths between speakers, was first noted by Helmholtz, but was shown viable during the Rutgers summer workshop in speech recognition by Andreou, Cohen, and Kamm in 1994.

From Knowledge-Based to Data-Driven Systems

Creating the underlying acoustic, language, and understanding models for speech applications has historically been a task of finding a set of rules that maps distinct features, such as spectral features, key phrases, etc., into predefined classes, such as words, phonemes, etc. For example, to map acoustic cues into digits, *Davis* et al. [34.2] provided a set of rules that relate filter-bank spectral features into vowels; to train a language model for speech recognition, a set of rules was identified in the form of regular expressions that describe all the possible variations on how users may be expected to interact with the application; to build language understanding models, *Gorin* et al. [34.15] provided a method for routing calls by mapping a set of phrases into meanings. If a user says 'I want to register', for example, then the salient phrase 'want to register' maps the input request into a target class *registration*, enabling the caller to register [34.16].

Although knowledge-based systems have led to a series of successful applications including directory assistance and travel reservation, these systems have been found to be both brittle, since they reflect what the designer is expecting the user to say, and expensive to create and maintain, since they require extensive manual effort to craft the grammars and rules. Over the past two decades, there has been a migration from rule-based systems to data-driven, or statistical, systems. Data-driven systems rely on machine learning algorithms to model a set of features into a set of classes. Therefore, the models can reflect what the users say as opposed to

what designers believe what users may say. Statistical models have been shown in both government applications and large-scale industrial applications to be both robust, since they provide a rich representation for modeling various effects, as well as efficient, since they are easy to build once data becomes available.

The migration to statistical models led by the availability of speech and textual data has revolutionized speech recognition and understanding research. The famous saying *there is no data like more data* has been a key driver for improved speech recognition performance. State-of-the-art recognizers are currently trained with over 1000 hours of speech and several tens of millions of words, compared with just a few hours of speech over a decade ago. This migration to statistical models has also led to a new wave of advanced speech and language processing applications. IBM and Dragon [34.17] captured the speech dictation market in the 1980s by demonstrating speaker-dependent systems based on stochastic language models and hidden Markov models that can recognize several thousands of words. AT&T began a trial in the mid 1990s which applied statistical classification models for contact center call routing [34.18]. Companies such as *Nexidia* and *Verint* [34.19, 20] are today providing solutions for contact center analytics, applying statistical models for audio search of large volumes of conversational speech.

The State of the Art

Over the past decade, there have been numerous studies on improving speech recognition accuracy [34.4,21–23]. Better training and adaptation methods have been on the forefront of research. Several research institutions have begun to apply discriminative approaches that can indirectly reduce the word error rate as opposed to the traditional maximum-likelihood methods that aim to increase a likelihood function. Examples of such methods include linear discriminative analysis (LDA), which is essentially a linear transformation of the acoustic features (or the model parameters) in such a way to increase the ratio of inter- to intraclass distance. Other methods include MCE and MMI training methods. These discriminative model training methods have been fairly successful in improving word discrimination by increasing the separation between the correct hypothesis and an alternative hypothesis based on N-best methods or some form of a word lattice [34.6].

Although word accuracy has been the most dominant measure used to evaluate speech recognizers, deploying speech recognition engines typically imposes several other technical challenges. These challenges, which have been the subject of significant research over the past two decades, are detailed below.

Efficiency. Having faster decoders with a small memory footprint is essential for supporting large-scale speech recognition applications in the network and limited vocabulary speaker-dependent speech applications on handheld devices. Advances in this area have been possible due to a numerous set of algorithms including the use of finite state transducers that enable efficient representation of the recognition network through general composition and determinization algorithms in both time and space [34.24]. Other methods include those that attempt to reduce the likelihood computation by minimizing the number of Gaussian computations. Such methods include Gaussian selection [34.25], which works by ignoring the Gaussians that contribute minimally to the state likelihood.

Robustness. Creating speech recognition systems that are invariant to that extraneous background noise, channel conditions as well as speaker and accent variations that are inherent in large-scale deployments of speech services in a major challenge and has been the subject of significant research over the past two decades. The basic problem is that adverse conditions as well as speaker differences cause deterioration in recognition performance. For example, applying models trained on wire-line speech to wireless applications typically results in 10–20% absolute degradation in word accuracy. This level of degradation causes many applications of speech technology to be unusable. Advances in this area include the use of cepstral normalization methods, such as cepstral mean subtraction or signal bias removal [34.25], which aim to remove the component of the cepstrum that is mostly sensitive to channel variations. Other methods include vocal-tract-length normalization (VTLN) [34.22], which reduces speaker differences attributed to variations in their vocal-tract length. VTLN works by computing a factor that enables compression/expansion of the spectrum. In theory, this is equivalent to downsampling/upsampling the speech signal. Other robustness methods include maximum-likelihood linear regression (MLLR), which adapts model parameters towards a speaker or an environment.

Operational Performance. Although many speech recognition applications demand a trade-off between speed and accuracy, the operational performance of these applications depends on additional measures such

as latency, end-pointing, barge-in, rejection, and confidence scoring. These measures affect the application user experience. Barge-in, which is based on using statistical methods to determine whether speech is present or not, enables users to speak while the prompt is playing. Rejection and confidence scoring provide measures of *goodness* of the recognized speech. These methods are essential for rejecting out-of-vocabulary words and for providing the application with a reliability measure of the recognition output. Although methods for improving the operational performance of speech recognition applications have traditionally been rule based, state-of-the-art systems rely on statistical models to provide an additional degree of robustness.

Multilingual Processing. The migration of speech recognition technology towards statistical methods has resulted in portable methods for multilingual processing. LIMSI's research shows that methods that work well for English seem to work just as well for other languages, especially for European languages. Nevertheless, achieving robust high-accuracy multilingual speech recognition requires continued research to handle limited training data, to deal with tonal languages, and to provide word pronunciation dictionaries. These advances are essential to provide both industry and the military with necessary technologies to support various applications including surveillance, audio mining, contact center automation, and speech translation [34.26].

Learning. The availability of large amounts of data is a key driver for the timely creation of cost-effective speech-based applications. This data, whether in audio or text form, is not only an essential source for improving the underlying speech and language technologies, but it is also critical for identifying problematic areas within a service that may need immediate attention. Logging, transcribing, storing, and managing this data, however, is often one of the most difficult tasks when scaling voice-enabled IVR services.

As technologies continue to transition to becoming more statistical, relying more heavily on speech data as opposed to knowledge rules, the quality and transcription accuracy of this data will continue to play a central role in the success of deployed services. In addition, improving the speed at which a dialog service is developed, deployed, and maintained is essential not only for scaling of these applications, but also for providing product differentiation.

AT&T has created a general learning framework based on daily feeds of speech service data to enable them to perform three learning operations [34.27, 28]:

Active Learning. Minimize the effort needed to label data by automatically identifying only those interesting or problematic dialogs that, when manually transcribed and labeled, would result in a significant improvement in system performance.

Active Labeling. Automatically identify inconsistencies in the manually labeled data. This process is applied for detecting outliers and incorrectly labeled data.

Active Evaluation. Automatically tracks trends in the performance of voice-enabled services to generate automatic alerts to indicate when to update or adapt a service.

In addition to these research advancements that have enabled widespread deployment of speech recognition applications, it is important to note that an additional important factor in creating usable speech applications is the user experience (UE) design [34.29]. UE can be considered as the technology front-end and plays an essential role in the success of voice applications, especially during problematic situations caused by system failures or by users not providing concise or accurate information. Poor UE design can very easily overshadow the best technologies. Conversely, good UE design can make up for deficiencies in poor technologies. The combination of statistical and robust methods for voice recognition and understanding as well as good UE design is what results in natural and intelligent voice applications.

34.2 Network-Based Speech Services

34.2.1 The Industry

Little did Alexander Graham Bell and Thomas A. Watson know in 1875 that the *harmonic telegraph* would spawn industry after industry, revolutionizing the way people do business.

One such broad industry is interactive voice response (IVR) systems for business call centers and consumer

services, which provides an automated communication mechanism between people (customers, employees, suppliers, partners, stakeholders, etc.) and an organization (businesses, enterprises, government agencies, etc.) to accomplish or facilitate a task (customer care, information request, and access to messaging and telephony services such as voice dialing).

Initially dominated from the 1970s through the 1990s by telecommunication and computer giants worldwide (e.g., AT&T, Lucent, ITT, Nortel, NTT, BBN/GTE, IBM), starting in the 1980s small niche startups that were nimble and singularly focused began to emerge in the marketplace. The characteristics of these companies were that they could react more quickly to market demand than big industry could. The main focus of these companies was to provide simple voice-enabled services to the call center industry. Voice automation of call center services and old-fashioned touch-tone-based IVR services began to increase their acceptability in the marketplace.

The industry has now matured into a highly diversified *food chain* that encompasses hundreds of equipment suppliers (for example, automatic call distributor (ACDs), private branch eXchange (PBXs), media servers, routers, phones, PCs, application servers), software suppliers (for example, operating systems, browsers, agent management systems, routing engines, speech engines, and computer telephony integration systems), system integrators, service providers, application developers, and user-interface designers.

The first voice-enabled service, automatic answer network system for electrical request (ANSER), was offer by Nippon Telephone and Telegraph (NTT) in Japan [34.30]. The service was deployed in 1981 and used very small-vocabulary isolated-word speech recognition to automate customer access to banking services typically handled through personal contact with human agents. Today, it has been estimated that US companies spend well over 120 B $ annually on business calls centers to deliver services to their customers, suppliers and employees. According to the Yankee Group, approximately 80% of that cost is spent on agent personnel costs, including direct expenses such as salaries, and indirect expenses such as benefits, administration, and buildings (Fig. 34.1). Automating even a fraction of the calls currently handled by live agents will generate tremendous cost savings.

On the consumer side, network services began entering the marketplace in the 1980s, enabling people to better access information and personal communications services. Since then, voice-enabled personalized communications services have expanded to include voice dialing, voice access to messaging (both voice mail and e-mail), and access to general information such as news, weather, sports, stocks, directory assistance, etc. In today's web-based nomenclature, we could call this group of services – *voice portal* services. These services provide consumers with simple, easy-to-use access to the services and information they use everyday from wherever they are.

Seizing a real business opportunity for speech technologies, many companies rushed into the market to provide speech recognition engine components. Such companies included, Verbex, Threshold, Centigram, Kurzweil, L&H, Interstate, Scott Instruments, Phillips, VCS, and Dragon. These products sold for hundreds and even thousands of dollars per engine, and were capable of recognizing isolated words with vocabularies ranging from two to a few thousands of words. While services based on speech technologies generated large cost savings for businesses that deploy them, few, if any of the speech vendors generated profits from selling speech engines. Therefore, over the years most of these companies merged with other companies or just disappeared.

In the mid 1990s, during the Internet boom, two new speech vendors came to the marketplace that provided much needed energy and marketing savvy to the industry – Speechworks (started as *ALTech* by speech scientists from MIT), and Nuance (started as *Corona*, a spin-off from SRI). As these companies grew, so did the voice-enabled services industry as a whole. Their focus, as with the early pioneers, was on providing speech technologies to the call center business market. However, both companies realized early on that providing only speech engine products to their customers was not a sufficient business model. While speech science has come a long way, there was still much to do to support a viable industry. In particular, the process of creating a voice-enabled service that is acceptable to consumers today is as much an *art* as a science. Hence, both companies invested heavily on providing a wide range of service development and monitoring tools, reusable service templates, and professional services (e.g., user-interface design support) that helped their clients build and maintain world-class services. This was, and continues to be, absolutely essential to the widespread success of speech recognition applications within the business community.

Over the decade since their entry into the market, Speechworks (latter merged with ScanSoft) and Nuance have been responsible for hundreds, if not thousands, of successful network-based voice-enabled services. While both of these companies succeeded greatly in helping

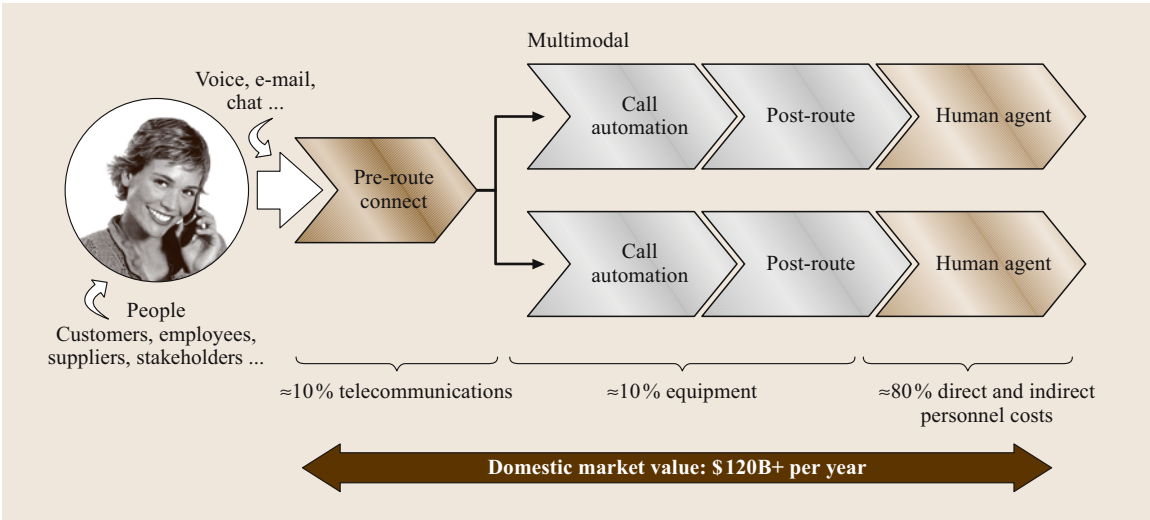


Fig. 34.1 Call center costs

make speech recognition a household name, in 2005 both were merged into a single company – called Nuance [34.31].

Over the past decade the market has differentiated further with the creation of companies that host services, but do not create or sell speech engines or platforms (e.g., Tellme , beVocal, TelUreka, Convergys, and West) [34.32–36]. These companies are very flexible in meeting market needs for quick service creation and deployment, yet are completely dependent on others (e.g., Nuance, IBM, Microsoft, and AT&T) for speech technology innovation.

34.2.2 The Service Paradigm and Historical View of Service Deployments

It is important to bear in mind that today the speech technologies described above and elsewhere in this Handbook remain error-prone and will remain so for the foreseeable future. For this reason, successful network-based voice-enabled services have been, and continue to be, those that have the following important characteristics.

Simplicity. Successful voice-enabled services are natural, easy to use, and intuitive to the user. Additionally, they provide real value to the businesses that deploy them.

Evolutionary Growth. Most applications introducing speech recognition were extensions of pre-existing IVR services. For example, speech recognition as a touch-tone replacement for IVR menus. Therefore, the users were familiar with the services and generally accepted them.

Tolerant of Errors. Given that speech recognizer will make mistakes, the inconvenience to the user must be minimized. This means that careful design of human–machine interaction, (e.g., the call flow) is *absolutely* essential in providing reliable and acceptable services. Hence, a demand for skilled user interface designers has occurred over the past decade.

This being said, there are some general questions that one must ask when considering an application using current speech technologies. The answers to these questions will help determine whether it is advisable or possible to design a quality application using speech technology. Some of these questions include:

- Are the potential users predominantly *power or novice users*? Power (or repeat) users generally desire a very simple design, i. e., the customers wants to get in and get out as fast as possible. Examples, in this space would include stock quotes and voice dialing. However, novice users require hand-holding (e.g., more instructions) and error-handling capabilities to enable them to achieve their goal effectively. This would be the case for most business-related customer-care IVR services.
- What environment will the user be speaking in (e.g., a noisy airport, quiet home, VoIP on a PC)? While advances to the robustness of speaker recognition systems have occurs over the past decade, robustness

still remains one of the key challenges for speech scientists today.
- What performance will a user of this service tolerate, and is speech recognition accurate enough to meet those expectations? This is critical in setting the expectations of all stakeholders, e.g., the business deploying the service and the customer using the service.
- Does the benefit of using speech technology in this application outweigh its cost, compared to alternative technologies? This is obviously critical in developing viable business cases for deploying a voice-enabled service.

Over the years, as more and more voice-enabled services have been development and deployed, speech scientists and engineers have gained a much richer understanding of the way people need to use speech technologies to positively affect the usability of many network-based services. For example, when people call a call center, there is a problem that they want solved or some information that they want. In addition, they often do not even know the *name* of the problem they are trying to solve. They generally only know the symptoms of the problem – for example, *I got a new bill and there is something on it that I don't understand*, or *I want instructions on faxing prescriptions in to a pharmacist*, or *Can I claim my daughter as a dependent on my tax return?* So, when we think of truly automating services with speech technologies there are two necessary goals that must be achieved:

- Determining *why* the customer is calling,
- Fulfilling the request to complete a transaction.

From the 1980s to today, given the state of the art in speech technologies, addressing the *why* question was generally achieved by asking the caller to provide the answer by navigating (via spoken command words or touch tones) a sequence of menu choices. In these system-controlled *directed dialog services* (i. e., guiding the caller step-by-step for information), the burden is placed squarely on the users to convert the thoughts in their mind to the physical actions presented to them at every level of the IVR menu. Users had to correctly navigate through a maze of complex menus to get the service they want. That is, users had to tailor their interaction to fit the capabilities of the service and technologies, not the other way around.

The first class of rudimentary or primitive directed-dialog-based applications that began to appear in the late 1980s and early 1990s were those services that used speech technologies to replace functions that human operators/agents carried out within call centers, thus reducing the cost of doing business for companies. Such *low-hanging-fruit* applications were able to make use of the then state-of-the-art capabilities in small-vocabulary isolated-word speech recognition techniques, and innovations in word spotting and barge-in technologies [34.37,38] to provide value to businesses – saving billions of dollars by reducing costs of providing services to customers. Examples of such pioneering applications included:

- Automation of operator services, deployed by many telephone companies including AT&T's voice recognition call processing (VRCP) service (1992) [34.39], Northern Telecom, Ameritech (now AT&T) and New Jersey Bell (now Verizon),
- Automation of directory assistance by Nynex (now Verizon) and Northern Telecom
- Control of network fraud developed by Texas Instruments (TI), Sprint and AT&T,
- Automation of banking services offered in 1981 by NTT in Japan,
- Reverse directory assistance in which a user enters a telephone number to retrieve a name and address, provided by Ameritech/AT&T and Bellcore (now Telcordia) in 1992,
- Information access services, such as a stock quotation services trialed at BNR in 1993.

Throughout the mid 1990s and early 2000s, network-based voice-enabled services established themselves as mainstream in the IVR industry. Applications that aimed to reduce the cost of doing business have completely dominated the industry and provided much of the justification for continued research and development for speech technologies. Over this time period, advances in processing and memory capabilities and in speech technologies themselves have enabled new applications to be deployed that require larger vocabularies (i. e., thousands of words), complex grammatical structures with continuous speech recognition, and more-complex directed dialog design. Such applications, which have provided real value to both business and their consumers, are now pervasive throughout the IVR industry in several business sectors including:

- Telecommunications
- Banking, finance, and insurance
- Retail sales
- Energy
- Education

- Travel
- Healthcare
- Government

Across these industries there has been some common themes on the types of applications deployed that provide both a value-added service and that can be supported given the limitations in capabilities within speech technologies today. This commonality has brought a high degree of reuse of knowledge across industries, and thus has enabled quicker and lower-cost development of voice-enabled services. Examples of horizontal applications that cut across industries include:

- Customer care routing (i. e., routing a customer's call to the proper subapplication via speech commands)
- Corporate and white-pages directory assistance (e.g., speech-based *411* for name and address listing retrieval)
- Corporate information access (e.g., account balance, order status, flight arrival/departure information, tracing shipments, locating retail stores, resetting passwords)
- Personal information access (e.g., voice dialing, access to messaging, stock quotes, news, weather, sports)

34.2.3 Paradigm Shift from Directed-Dialog- to Open-Dialog-Based Services

Directed-dialog services have been the mainstay of the voice-enabled services market since the mid 1980s. The main characteristics and assumptions of successful directed-dialog-based services have been:

- They consist of a small number of easily understood user options with a simple easy-to-follow call flow.
- The customer is not vague with his/her spoken input.
- The application does not need to combine statements of intent with a lot of additional information within the same utterance (that is, extra information that goes along with an intent, for example, *Gee, I'm having a problem with my March bill*).

However, directed-dialog design imposes several limitations on network services that are obviously designed to work for millions of consumers. The biggest is that people tend to use their own words or phrases in describing a problem, and they are often vague in their request and need the opportunity to clarify their intentions. They sometimes change their minds in the course of a dialog, mix all kinds of information together, and their meaning often depends on the context of where there are within the conversation. In the end, depending on the specific service, a business's end-users *self-misroute* themselves through their existing directed-dialog-based IVR 20–60% of the time. Simply stated, people find it hard to navigate menus. Users cannot change their minds/context, recover from mistakes, skip steps, or add relevant information. In the end, customers are frustrated. This should not be a surprise to anyone who has used an IVR service. So, while directed-dialog-based services have been adequate to create the voice-enabled services market over the past couple of decades, they are too inflexible and limiting to allow the market to move forward and become ubiquitous in network applications.

In the 1990s, the Defense Advanced Research Projects Agency (DARPA) began a program to support open-dialog research. The air travel information system (ATIS) was an extensive project aimed at extending our knowledge in speech recognition for *simple* services to those requiring more-complex interactions. Universities, such as MIT and industrial labs such as BBN, IBM, and AT&T were the first to use natural-language understanding and simple dialog management to enable services to have simpler, easy-to-use interfaces to services. Research in open-dialog-based call routing services, led by AT&T, over the past decade has focused on spoken natural-language understanding (NLU) and dialog management (DM) technologies for call classification, whose function, along with large-vocabulary speech recognition, is to determine first, *why* a person is calling, and second to fulfill their needs through a conversation. The innovation of open-dialog-based services enables a natural interaction between a customer and an automated service, therefore simulating human-to-human-like interactions. That is, open dialog services shift the burden of determining how to use a service from the user to the service and associated speech technologies. These technologies, while challenging to the speech and AI Research communities, will enable the voice-enabled market to explode – breaking through the barriers created by menu-based services.

AT&T's research formed the basis for AT&T's customer care system called *How May I Help You* (HMIHY), which was deployed in the network initially in 2000 by AT&T consumer services [34.18]. The HMIHY service, when fully operational throughout the US, handled about 3.5 million voice dialogues with customers every month. This service was the first open-dialog spoken natural-language system to be fully deployed in the world. It significantly improved cus-

tomer satisfaction as represented by four measures, namely:

- Decreased repeat calls by 37% (customers who cannot navigate touch-tone menus hang up and redial)
- Decreased the rate of customers switching to another long-distance company by 18%
- Decreased customer call agent times by 10%
- Decreased customer complaints by 78%

Following this success, in 2002 AT&T launched its VoiceTone IVR service, which took advantage of the new advances in natural-language understanding and dialog management. Since then, many open-dialog-based services have been developed and deployed. Successful characteristics of these services fall into several categories, including:

- Automation of new customer transaction services that previously had been *off-limits* due to limitations in traditional menu-based touch-tone and simple speech recognition technologies.
- Increased *self-service* completion rate.
- *Improved navigation and content delivery* and a reduction of misdirected calls, as well as substantially shortened call times.
- A reduction in customer callbacks resulting in increased customer satisfaction. This will positively impact the client's ability to meet their objectives and improve their customer retention rates.
- Customer perception of a more-personal experience through the use of both external sources of information (for example, account information and attributes, offer information and attributes, and business-targeted outcomes and goals) and through flexible interaction with the customer (for example, contact conditions and attributes, caller intent, and response).

The following interaction is an example of an open dialog interaction possible through VoiceTone.

VoiceTone: *Hello, this is Barney Health Care, how may I help you?*
User: My medicine ran out last week.
VoiceTone: *Would you like to refill your prescription?*
User: Yep uh my refill I.D. number is PB14720.
VoiceTone: *Okay, I have ordered for you a new prescription. Is there anything else I can help you with?*
User: No thanks.
VoiceTone: *Thank you for calling Barney Health Care. Goodbye.*

34.2.4 Technical Challenges that Lay Ahead for Network-Based Services

History aside, the true vision of automating network-based services is to provide the user with access to services independent of the user's communication device or location. That is where speech technologies become necessary. Advances of speech technologies in several dimensions have been and will continue to be essential to the viability of deploying automated services in the future. That is, *we have not solved the problem*. Amongst the key challenges that must be addressed are

1. speech technologies are still too fragile to use
2. too expensive to deploy
3. too hard for developers to use to design network services

Improving Performance. Performance is important not only for meeting the business success criteria but also for ensuring high-quality customer experience.

In the area of performance, improving the robustness and accuracy of speech recognition, the breath of natural-language understanding beyond call classification, the naturalness of text-to-speech synthesis, and the flexibility of dialog management are subjects of current focus. This area represents the bulk of the research over the past few decades and is important for meeting business success criteria and ensuring high-quality customer experience.

Speeding up Service Deployment Through Automation and Reuse. Automation is critical for reducing the development effort during both creation and maintenance of these services.

For increased automation, novel self-learning algorithms for reducing overall development resources during the creation of natural-language applications are being applied. In particular, they use the *active* family of methods, which includes *active learning* for reducing transcription and labeling effort, *active labeling* for ensuring high-quality labeled data necessary to create natural-language understanding models, and *active evaluation* for evaluating and adapting models automatically with no human intervention. Automation in these technical areas is critical for reducing the development effort during creation and maintenance of these services.

Increased Scalability. Scalability is a key component of the mass deployment of cost-effective natural-language dialog services. Improving the efficiency of speech algorithms and their implementation (i. e., speed, memory, and size), leveraging standards, and exploiting reusable data and libraries is the focus of much research.

Adoption of Technology and Application Standards. Understanding the need to scale application development, in 1995, AT&T created the phone markup language (PML) application program interface (API) for rapid creation of interaction voice response services. In 1999, PML was released with the founding of the voice extensible mark-up language (VXML) forum by AT&T, Lucent, Motorola, and IBM, responsible for developing and promoting the VXML standard currently used industry wide. Today, VXML is the standard environment for development of most IVR services worldwide. Going forward the W3C voice browser group is working on expanding the capabilities of VXML to include support open-dialog services, and especially services that require multimodalities of the input devices, such as voice, touch, writing, etc. This will drive the effort of unifying network-based telecommunications services with web- and device-based services into a single environment that makes it easy for application developers to create new and innovative services [34.40, 41].

Additionally, standard interfaces, such as the media resource control protocol (MRCP) [34.42], that allow for rapid integration of new speech engines into service delivery platforms have begun to emerge into the marketplace. Such standards enable improvements and advances in speech technologies to find their way more quickly into the field.

34.3 Device-Based Speech Applications

34.3.1 The Industry

Portable devices such as cell phones offer a unique opportunity for speech recognition technologies, either by themselves or as part of a multimodal interface. If these devices communicate wirelessly, they tend to support both IP networking and standard cell-telephone communications, and thus the devices support both telephone interchanges and web-based data exchange.

It has always been challenging to design an interface on a small device that enables fluent use of the device, while allowing full access to its capabilities. For a cell phone, the 12-button interface is a copy of the touch-tone phone keypad, which itself is a representation of the 10-hole *dial* ring of the original telephones. This interface was designed in a time when the phone number was the only critical information that the system required, and it was very efficient at delivering that information to the network.

With the advent of new services, a user often wants to input text, video, pictures, or other nondigit information. While the keypad is useable for many purposes, typing a *7* four times to produce an 'S' is tedious for all but the most dedicated teenage text messenger.

The interface on a standard PDA device is even more limited, although it can be assisted by the touch-sensitive screen, which often accepts handwriting or stylus inputs. It is possible to build a completely graphical operating system on this type of device, and early devices were button-poor boxes with touch-sensitive screens. While these devices were successful in the market as personal information managers (Palm controlled this market for many years), new devices have become less successful because they violate the *one-hand* usability criterion for mobile devices – that is – if you cannot use it with one hand, then you cannot use it while walking, driving, or carrying something, and thus the device is much less useable than it would otherwise be.

Speech has always been a natural command-and-control information exchange, at first between people, and then for people to control machines. This approach has come late to mobile devices because of the limited capabilities of the devices, the abominable design of early PDA acoustic systems, and the limited insight of the technological community.

34.3.2 The Device-Based Speech Application Marketplace

Embedded speech recognition is a relatively recent phenomenon. The markets for these devices range widely, but are evolving as devices evolve.

Early applications were mobile-phone dialing. AT&T was instrumental in designing and fielding a speaker-dependent digit and name recognizer with their automobile-based mobile telephone in the mid 1980s [34.42]. While the design included both digits

and names, the devices which were built and marketed contained only name recognition. The adventure was not a marketing success.

Mobile phones (remember the *bag phone*?) were relatively rapidly supplanted by smaller cell phones in the late 1990s and early 2000s. As computation became possible in smaller platforms, and battery technology improved, it became possible to build digital cell phones that were *luggable*, and, in the past half decade, which have become pocket devices. These communications instruments now have substantial computing capability, and are often multiply connected, using the digital cell-phone channel, a side-channel for text (SMS), and often a separate IP network and even a wireless fidelity (WIFI) network. While early cell phones had limited capacity, as their computing capabilities improved, speech researchers discovered that dynamic programming methods allowed digit and name recognition to be integrated into these devices [34.43]. These techniques were offered by AT&T, Motorola, NEC, Siemens, Qualcomm, and many other laboratories. This plethora of *speaker-dependent* applications were widely deployed, but failed the market usability test, as they were too complicated for the blinking-12 generation [34.44]. Name recognition for voice dialing is embedded in more than 40 million cell phones, and is widely ignored by the user population.

Since 2002, speaker-independent recognition has been available on many mobile phones, and it is now available on close to 100 million devices. This technology allows speakers to simply pick up and use digit and name recognition for dialing and command-and-control applications on cell phones. It meets the requirements for simple and robust applications, and has become a common application used by technological sophisticates. Microsoft offered both AutoPC and PocketPC speech solutions, but neither achieved traction in the market place. The PocketPC phone voice command became the number-one application seller for the CE platform, but that fact only attests to the small market which was served. It appears that the population at large has yet to discover this capability in their devices (a failure of marketing), and thus this modern technology is only now gaining attention. Media review has been enthusiastic.

There are other markets for embedded speech technology, although most remain skeletal. Games and toys can sometimes benefit from a speech interface, and Hasbro and other toy manufacturers have fielded a series of small toys and plush animals that both respond to voice and talk to the user (in this class is a very capable R2D2 robot, which is being reissued in the mid-2006 time frame). These devices have not risen above the novelty level, and remain uncommon.

Navigation and automobile command and control is a relatively successful market for speech recognition on noncomputing platforms. The advent on On-Star in the US market delayed sophisticated speech recognition, as the system was fielded with a very crude discrete speech recognition system, and remained in that state for several years. Only in the past year has On-Star begun to field continuous speaker-independent recognition (from IBM). The market success of this venture will be interesting to watch, but is far from assured. Automobile systems (including navigation) use speech as output, but speech recognition has not yet become a common input.

PDAs would seem to be ideal for speech systems, but for several years the audio systems in IPAQs and other PDA devices offered audio quality ranging from inconsistent to abominable. (It is easy to mix audio and digital signals in computing platforms, and this is especially true of very small devices.) Continuous speech recognition was demonstrated on an IPAQ platform in 2003 by Voice Signal in English, and was simultaneously announced by Phillips for Mandarin. Neither company has fielded a commercial product on these devices for dictation, and the field remains unserved. (In the case of the Voice Signal demonstration, an outboard audio system was used to mitigate the IPAQ native systems.) Microsoft now offers a voice dialing system on its Windows CE systems that combine phones with PDAs.

The continuing increase in both memory and computing capability in cell phones allows dictation recognition in recent devices. Discrete large-vocabulary recognition in cell phones was announced in a commercial device (Samsung P-207) by Voice Signal in 2005, and continuous recognition was demonstrated in 2006. Nuance has also announced offerings, both embedded and distributed speech recognition (DSR)-server based. While text creation for email, messaging, and internal annotations seems an obvious requirement for communicating platforms, the market remains small. It is unclear whether communications (SMS), mobile search, location-based services, or some other application will ultimately become the market maker for this technology.

34.3.3 Technical Challenges that Enabled Mass Deployment

There have been many technical challenges to successful speech recognition in mobile devices, although some of these are being addressed by continuing technical advancement. Additionally, the rather limited

processing and memory resources that are available on mobile devices, compared to the seemingly unlimited resources available for network applications, imposes unique technical challenges for even the simplest mobile applications, let alone the complex applications that can be deployed within a telecommunications network.

Speech recognition algorithms range from the very simple – dynamic-programming-based speaker-independent small-vocabulary recognition systems – to the extraordinarily sophisticated HMM-based large-vocabulary continuous recognition.

Dynamic programming may be used to create an *acoustic pattern matching* algorithm, as described in *Sakoe* and *Chiba* [34.43]. While this pattern matching works well for a single speaker in a consistent environment, it produces a brittle matching algorithm that fails in moderate noise or for other speakers. On the other hand, the basic algorithm simply captures the speech to be modeled, derives a straightforward time-based model, and allows this model to be compared with a later spoken utterance. Because it is an acoustic matching algorithm, and must be trained by the user, the initial memory cost is very low. However, because the algorithm knows very little about speech or the signal that it is to model, the reference models tend to be memory intensive. Thus these algorithms can run in small-memory systems (as they are also not very compute intensive), but work only for small vocabularies (digits or 10 names, in typical applications).

Early implementations of dynamic programming algorithms used the digital signal processing (DSP) computation engine that is part of many cell phones. There was ample computation, and the low memory requirements allowed name recognition. Sensory announced toolkits for these technologies in 1997, and software for DSPs in 1999. Major cell-phone handset manufacturers (Nokia, Siemens, Motorola, NEC, Qualcomm, and others) supported internal speech laboratories that produced similar software throughout the 1980s and 1990s.

Modern cell phones have two separate processing systems – a digital signal processor and a second which is the user interface (UI) processor. The market leader for the UI processor is ARM, which first was designed in 1985, and has become dominant in the field. Modern ARM processors run at speeds from tens to hundred of MHz. In cell-phone systems designed in 2005 and 2006, these ARM processors can support applications independent of the cell-phone functionality of the devices, and often allow after-the-fact programming using BREW, Microsoft CE, or JAVA.

The UI processor has become a sophisticated platform in cell phones. While the DSP has become a closed system dedicated to cell-phone functionality, the user interface processor has become an *open* computing platform. Memory in modern cell phones ranges from 16 to 500 MB. Thus, these UI processors have computing capabilities ranging from those of the 286 to those of the Pentium I, except that they do not do native floating point. Audio systems in cell phones, with the exception of Bluetooth systems, tend to be well designed, and microphone placement is often optimized to minimize mouth sounds and extraneous noise. There are no remaining technical constraints on the implementation of speech recognition applications in these devices. Successful products now require user applications that are intuitive and discoverable, but these requirements are essentially independent of the speech recognition technology.

In remote-control applications, speech recognition has had mixed success. Standard remote-control devices remain technically difficult for speech applications, as they are generally built around 4 bit or 8 bit processors. (This is sensible if battery life is the primary factor, as it is in these devices). Some advanced remote-control devices (see Sensory for examples) provide limited speech recognition input, but this remains a novelty market.

There has been an industry call for a general solution to the remote control problem for advanced television services, since the blinking-12 generation cannot manage video on demand, the premier cost-plus service in television. Research has been focused on array-microphone solutions centered on cable boxes or other devices, but solutions remain suboptimal. This arena remains in flux.

34.3.4 History of Device-Based ASR

Embedded ASR solutions emerged from the development of *word spotting* as a topic of government interest during the 1970s. The general idea was that one could spot words in running speech by simply building a model of the acoustic signal that was the word or words of interest, and then search incoming speech for that acoustic event. The signal processing was, in general, LPC based, and very efficient. The search algorithm was a generalized dynamic program, which worked by attempting to match a warped version of the speech sample to the incoming speech, starting at all possible start points, and aborting if the *score* became too improbable.

Wide use was made of the Itakura distance, which Itakura claimed was an information-based mea-

sure [34.45]. The computation of the matching distance could be made very efficient, and the task could be carried out in early signal processing chips. On the other hand, the acoustic models of the example speech were quite large, and thus in early cell phones using this technology it was only possible to search for 10 or 20 names, despite the much larger phonebook capabilities.

Voice dialing using the dynamic programming method was available in the mid 1980s on several wired devices. Among these was the Freedom Phone, sold by Southwest Bell in the 1980s, which had a 50 item dynamic time warping (DTW)-based speech recognition system and also a speaker trained digit system. About the same time, Bell Laboratories built a voice dialer (called the Victory Dialer) for a car phone which recognized names.

In 1987, this DTW technology was used by Interstate Electronics to build a car-phone dialer that would dial either names or digits. The digit dialer was speaker independent, and was trained from a small database of digits, while the name recognizer had to be trained by the user. This dialer was plugged into the handset connector on a car phone, and the handset was, in turn, plugged into the dialer. The technology was extremely effective for its time, and the company sold several thousand of these devices. Unfortunately, the *car phone* market (the radiophones) was quickly supplanted by cell phones, and the market for these dialing devices disappeared.

34.3.5 Modern Use of ASR

While DTW technology was used by cell-phone manufacturers and a few other companies (ART – Advanced Recognition Technology, an Israeli company, was the most effective private purveyor) throughout the late 1980s and 1990s, the field was overtaken by much more-robust HMM technology. In 2002, Voice Signal introduced the first HMM-based speaker-independent digit dialer in the A500, a Samsung phone, and a year later introduced speaker-independent name dialing in the A610. From that time onwards, HMM technology has proven to be an effective and robust method of voice recognition in cell-phone platforms, and has even been extended to dictation systems of many thousands of words. The two largest providers of speaker-independent voce dialing were ART (now part of Nuance), and Voice Signal. The market is just emerging, although more than 75 million cell phones contain this technology.

The conjunction of mobile platforms and speech recognition is beginning to have an effect on the assistive technology community. It is now possible to use cell phones with only a single button push, and small-footprint synthesizers allow the use of these devices by visually impaired people. (Some devices *say* the name of any button pushed.) This market is not well established in the United States, but there is government support for such devices in the European Community.

34.3.6 Government Applications of Speech Recognition

General Service Overview

Government interest in speech recognition and in speech processes in general, covers many facets of the industry. Among the most interesting and unique are the uses of speech recognition in command-and-control applications, low-bandwidth communications, and intelligence functions.

Speech recognition interfaces are now being developed for aircraft simulators, and are specified as part of the standard avionics in the Joint Strike Fighter. The continuing DARPA speech programs, largely concentrated on speech and language recognition and interpretation of broadcast and conversational information, can be expected to yield systems that make transcription and analysis of intelligence information easier in the future. Other programs, such as TRANSTEC, are attempting to create mobile platforms that assist communications between speakers of non-intersecting languages. While current devices are concentrated on Arabic and Mandarin, the technology is being created to be easily ported to other languages, and commercial opportunities are awaiting enough technical advances to make the markets financially rewarding.

Technical Challenges to Overcome for Deployment

Technological evolution has made state-of-the-art speech recognition available in many devices and processes, and the availability of memory and computing power follows that availability in the commercial space, delayed by a year or two.

This delay is due to a multiplicity of factors. For field use, systems must be able to work over a large temperature and humidity range, and must tolerate physical abuse found in combat situations. Among the current platforms of choice are the Panasonic Toughbook and others of its type (there is a MIL standard, 810, which specifies insensitivity to temperature, humidity, liquid spills, salt, gunfire, acceleration and other factors). While central processing unit (CPU) speeds tend

to lag the commercial space, they tend to catch up within a year or two, so any process available on a commercial laptop today can be expected to run on a hardened processor in a short while.

On the other hand, acoustic requirements for military systems are generally assumed to be extreme, with operating equipment (tanks or airplanes), and military field operations (gunfire, crowds, etc.) forming the most challenging problems. Speech systems for translation are now being tested in Iraq, and have generally met with a positive response, although the technical program managers tend to be very conservative, and system integration time is very long. Speech recognition solutions have been deployed experimentally in the US Navy, Marines, and Air Force, but large-scale deployments have not yet been made.

Initial Uses of Speech Technologies

Two interesting applications of speech recognition are in general intelligence applications and in low-bandwidth communications systems.

With the advent of competent large-vocabulary speech recognition and highly intelligible speech synthesis, a low-bandwidth communications process can be completed by simply recognizing the speech into words, sending the text, and reconstructing the audio at the other end from the word string.

For general intelligence applications, an effort has been underway for many years with (D)ARPA funding of large-vocabulary speech recognition for the Department of Defense. In its original instantiation, the DARPA project of the 1970s defined the goals of a computer-based speech recognition system to recognize continuous speech, using a vocabulary of 1000 words, in real time. This project, called the Speech Understanding Research program, was funded at CMU, Stanford Research Institute (SRI), MIT's Lincoln Laboratory, Systems Development Corporation (SDC), and Bolt, Beranek, and Newman (BBN), and became the basis for speech recognition research worldwide. This project and its follow on allowed Dragon Systems to develop commercial discrete, then continuous speech recognition systems first on plug-in personal computer (PC) boards, and then on the native PC platforms themselves.

DARPA has since funded several follow-on projects in speech recognition, and in 1993 they funded a summer workshop in speech recognition at Rutgers University, where recorded spontaneous speech was, for the first time, the target of the research community. These efforts were aimed at *intelligence* applications, where it was assumed that some entity would transcribe everything available to it and mine that information for intelligence purposes. Recently the focus of DARPA programs has been on applications of speech recognition technology including automatic translation and information extraction systems, in the global autonomous language exploitation (GALE) program. The intelligence applications of speech recognition remain the stuff of fiction and fantasy reporting, while commercial applications grow at a very slow rate.

Speaker Identification/Verification

Over the past few years, with the rise of international terrorism and the concomitant growth of automated financial interactions and identity theft, the government has begun to pay attention to the entire question of biometrics in personal identification. While biometrics is a larger field than speech recognition, the speech portion of a biometric suite has the ability to enhance system performance, while allowing identity to be assessed from audio signals. Audio is the one biometric that does not depend on having the identification target present for either system training, system test, or both.

The United States has fielded a handheld biometrics system for use in military activities, called handheld interagency identity detection equipment (HIIDE) (http://www.securimetrics.com/solutions/hiide.html), but this system does not yet contain speech recogni-

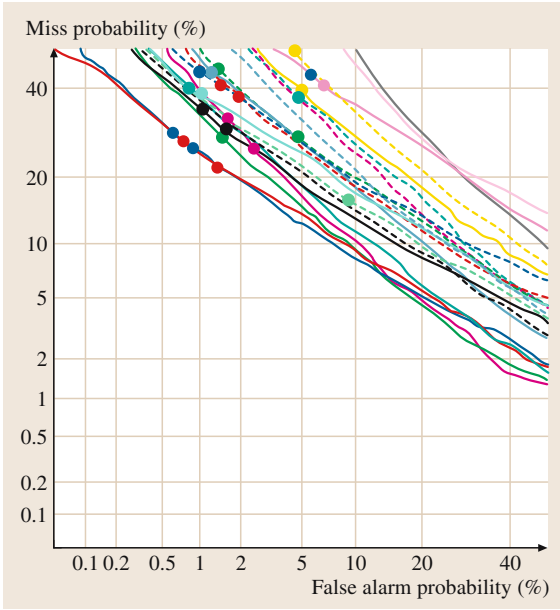


Fig. 34.2 Plot showing results from 21 systems for the 2002 primary condition in the one-speaker detection task

tion. In preparation for its inclusion in such systems, the government has been holding annual technology system performance evaluations with publicly available data and public results (although the identities of the participants are not part of the public results). In Fig. 34.2, one can observe the change in performance from 2002 to 2005 [34.46]. These improvements are due to more-sophisticated statistical modeling, increasing use of support vector machines (SVMs) rather than GMMs, and the extension of the scoring technology from short-term spectral analysis to long-term linguistically motivated features.

Speaker ID can be used in many ways, but the most common two are for speaker identification and speaker verification. While the technology improvements allow improved performance in both regimes, the open questions tend to be different.

Speaker verification assumes that the speaker's biometric identification (in this case, his voice) is accompanied by a claim of the speaker's identity. The task is to verify that the speaker is or is not the claimed person. This is the scenario of the National Institute of Standards and Technology (NIST) tests, where the error rates are computed over hundreds of thousands of trials. In this task, computation tends to be a secondary constraint (think, for instance, of an access system in which the talker speaks several words or sentences into a microphone, and then is granted access to a particular physical installation), while accuracy is crucially important. The acoustic channel is typically not an issue, since in most scenarios the speaker is a cooperative user of a system, which makes his life easier.

In the second scenario, one is typically attempting to do speaker identification (maybe speaker spotting) over a much larger collection of data, and the identity claim is missing. In that case, one must ask for each conversation whether the speaker is one of a known population of speakers, or whether the speaker is *someone else*. In this technology, the accuracy of the system tends to be less good, since the acoustic channel is not controlled (microphone, telephone, cell phone, etc.), but the task is more likely to be sorting data into interesting or uninteresting piles, and it may be assumed that secondary testing will be done on the output.

The reader can imagine more scenarios where speaker identification might be used, both in government and in industrial applications, and the technology is approaching usability for many of these.

34.4 Vision/Predications of Future Services – Fueling the Trends

Several technology trends and new technologies will fuel the growth in capabilities over the next few years. As the wireless network bandwidth pro capita and the speed of CPUs keep growing, rich and interactive multimodal and multimedia applications will continue to migrate from the core network to the periphery hosted by mobile devices, providing ubiquitous multimedia communication services *anywhere* and *anytime* to consumers and *road warrior* professionals. At the same time, broadband access is increasing at impressive rates and reached 65% of the American households by the end of 2005.

Although this scenario seems to suggest that video, data, and voice services are relentlessly converging to a seamless integrated internet protocol (IP) network, it is still unclear what kind of applications and standards will succeed in fostering next-generation human–machine communication services. It is certain, however, that these multimedia services, powered by a new breed of machines, will be operating globally supporting multiple languages, and will provide input access through a variety of different modalities (such as speech, gesture, pen, etc.). With this growth and complexity, improvements in service performance and deployment scalability that require minimal human intervention will be critical. Therefore, automatic learning techniques will become an essential component of these new services. Mining of heterogeneous information and data (call logs, speech, call center information, etc.) will also be an integrated part of these multimedia services to help extract trends and knowledge that can aid in improving business operation.

Predicating the future is always a risking task. That said, in this section we postulate several treads we envision seeing as the industry moves forward as it continues to mature into a mainstream market.

34.4.1 Multimodal-Based Speech Services

The widespread availability of broadband access, VoIP, and powerful mobile devices are fostering a new wave of human–computer interfaces that support multiple modalities, thus bridging and eventually blurring the device and network voice-enabled service markets. These

so-called multimodal user interfaces combine interactive multimedia output where audio, video, graphics, and text are accurately synchronized under time constraints, with several modes of input such as speech, keyboard, pointing devices, handwriting, and gestures. One example is the AT&T's multimodal access to city help (MATCH) multimodal service prototype (Fig. 34.3), which enables users to use speech, gesture, or the combination of the two modalities to access and retrieve information about cities and sites in the US [34.47]. This integration of speech and visual interfaces provide richer, more-versatile, and robust user interfaces than traditional speech input alone. Multimodal applications offer the ability to adapt the user experience to the actual terminal capabilities of the access device.

Although multimodal applications using mobile devices have received significant interest over the past five years, the central idea of using speech to manipulate a computer device goes back to the desktop dictation systems of the early 1980s. Several companies, including Dragon, Microsoft, and IBM, provided desktop solutions for speech dictation supporting speech, mouse, and keyboard input. These systems, which sold hundreds of thousands of copies primarily to the medical and legal industries, allowed users to click and talk. Speech recognition errors were corrected by clicking and either speaking or picking a word from a menu selection. Although early incarnations of these systems, which were essentially speaker dependent, required several training hours of speech and required users to pause between words, later systems that were available in the mid 1990s supported a much larger vocabulary set (> 5000 words), required only a few minutes of training data, and performed continuous speech recognition.

Fig. 34.3 AT&T's MATCH multimodal service prototype

Today, the Internet revolution and the availability of powerful handheld devices are facilitating the widespread of a new wave of multimodal applications. These applications support speech, gesture, and handwriting recognition, imposing new technical challenges to factor in speech recognition, natural-language understanding, and dialog management technologies as well as user-interface design, which is currently restricted to multimedia and video gaming applications. Major players in the industry such as Microsoft and IBM, have recognized the potential of this new multimodal market, proposing limited solutions to extend the concept of web browsing with speech input. Recently, Microsoft integrated its speech server into Microsoft Exchange 2007. Exchange is one of the most widely used messaging platforms for enterprises, thus positioning Microsoft well for providing flexible user designs for messaging services. Microsoft founded the speech application language tags (SALT) forum to promote voice applications as well as multimodal navigation in a web browser environment [34.48]. SALT interoperates with existing web standards, introducing a few new tags for speech control. IBM and Motorola, on the other hand, proposed and alternative approach to multimodal called XHTML + voice (X+V). X+V extends the existing Voice XML standards and complements them with a presentation layer built around XHTML [34.41]. In 2002, W3C launched a multimodal interaction working group with the purpose of finding common agreement upon a general architecture and an approach to define multimodal applications. The next generation of multimodal services will take advantage of integration of modalities to achieve robust performances.

Among the technology challenges when dealing with multimodal interfaces is the need to provide synchronization among the different input modalities. Several organizations, including AT&T, IBM, CMU, and IDIAP, have been actively pursing different strategies for synchronized modality integration. In AT&T's multimodal systems for information access to restaurants and subways, for example, users can circle with a pen, speak directly to a mobile device, and/or write on the screen. Mutual disambiguation, when using speech and gesture inputs simultaneously, is supported using a multimodal finite-state

transducer, which outputs a semantic lattice representation [34.47]. The lattice is interpreted by a multimodal dialog manager, which produces the next dialog action. A multimodal generator renders a graphical response with eventual animation together with a text-to-speech synthesizer. Multimodal integration can also be viewed as combing different streams, firing at different rates, using an asynchronous HMM. Whether using finite-state transducers or asynchronous HMMs to support multimodal interfaces, there is an inherent limitation in performance attributed to the lack of multimodal data. Rapid creation of multimodal applications with limited training data is a major technology challenge the research community is addressing today.

Multimodal interfaces will solve many of the current limitations of voice applications. They provide not only a mobile access to information but also a richer selection process through the visual interface. This visual interface significantly improves the user experience when dealing with complex applications such as maps that require special dialog strategies including disambiguation, clarification, relaxation, and constraining. Multimodalities provide several enhancements to a spectrum of applications including:

1. Information access and navigation: speech input alone for retrieving information about restaurants and travel directions, for example, can be significantly enhanced with a multimodal interface, supporting speech, gesture and global positioning systems (GPS).
2. Instant messaging: with the availability of powerful mobile devices, multimodal interfaces provide users with a richer and a more-flexible interface for communication using text messaging, clicking, and/or speaking into handheld devices.
3. Meetings and conferencing: multimodalities provide users engaged in meetings and conference calls the ability to geocollaborate and plan on the same project or event.
4. Gaming: combining gestures, pointing, and speech facilitates new types of video games in which players can speak and gesture simultaneously to perform a task. It will also aid people who have dyslexia (difficulties with movement) and dysgraphia (difficulties with handwriting) to apply speech as a primary modality to play games.
5. Diagnostics: combining speech and gesture provide an easy-to-use interface to diagnose products and services, such as airplanes.

34.4.2 Increased Automation of Service Development Process

The task of creating customized spoken dialog applications is traditionally labor intensive, requiring significant data resources and A tremendous level of expertise. As a result, despite efforts to modularize and reuse components of the dialog, it is not surprising that only a few hundred speech services were actually deployed in 2003 for large business customers. These services tend to be highly customized and are typically designed independently of any other sources of information, such as a website, or human–human conversational data.

AT&T has been pursuing a new research direction to scale this industry by completely automating the process of creating spoken, or chat-based, dialog applications by leveraging the wealth of information on business websites. Given that most businesses maintain a website, the goal has been to leverage that information to design, create, and maintain a new line of automated services requiring no human intervention. A prototype system of these new types of services (WebTalk) has been created by AT&T. WebTalk employs speech, language, and machine learning technologies to enable customers to enter into an automated chat-based or a spoken-language-dialog-based *conversation*. It includes a website analyzer that automatically downloads web pages, analyzes them, and constructs dialog-oriented task knowledge.

Information extraction is a key component of the AT&T WebTalk system. It enables the system to extract key entities, such as a list of products and services, contact information, definitions, questions/answers, etc. This information, along with a general-purpose DM, enables WebTalk to exchange in a natural-language dialog without actually involving a person in the loop either during the creation or the maintenance process.

34.4.3 Complex Problem Solving

At the heart of a spoken dialog system is the *dialog manager*, the component which decides what the system should say at each point in the conversation. Whereas all of the other elements of a spoken dialog system, such as speech recognition, language understanding, and text-to-speech, are based on probabilistic methods and optimized based on data, dialog managers have historically been handcrafted by user-interface experts. Because speech recognition errors are common, this design task is extremely complex and, in prac-

tice, the dialog manager is often *the weakest link* in a spoken dialog system, causing conversations be either be very structured or to go wildly off-track after even a single speech recognition error. As a result, deployed telephone-based spoken dialog systems have been limited to highly structured tasks where user's behavior can be constrained effectively (such as banking or simple travel transactions) or can be predicted accurately using natural-language understanding technologies for call routing.

A family of new approaches to building dialog managers seeks to extend the reach of dialog systems by improving the ability of the dialog manager to cope with speech recognition errors, hence better enabling the automation the class of more-complex problem solving services. Conventional approaches make binary accept/reject decisions about each speech recognition result, and use accepted speech recognition results to form a single hypothesis for the dialog state. Inevitably, a speech recognition error will be accepted into this dialog state, which can cause a system to get *stuck* and lead to an unnatural and unproductive conversation. New techniques in development today are aimed at representing the state of the conversation as a Bayesian network, which allows three important improvements [34.49,50]. First, a Bayesian network maintains *multiple hypotheses for the dialog state*: every speech recognition result is viewed as having some probability of being correct, and these probabilities are aggregated over the course of the dialog, preventing the system from getting *stuck* on a single hypothesis. Second, this aggregation process can incorporate models estimated from data of how users *actually* behave, rather than designer intuitions about how users might behave. Finally, automated planning in the form of partially observable Markov decision processes can be applied in a principled manner to ensure that the system always chooses an optimal action.

For example, consider the troubleshooting domain, in which a user is trying to restore a faulty product or service (such as an Internet connection) to a working state. Dialog systems in this domain today are highly constrained, for example asking a series of yes/no questions. While this approach avoids engendering more-complex speech, a forced regiment of questions can be tedious for callers who can feel trapped when a system cannot understand, for example, *But I've already tried rebooting my computer and that didn't work*. To build better dialog systems in this domain, more *mixed initiative* capabilities are required, which allow users to take control of the conversation, offer new information, or revisit an earlier part of the conversation.

Advanced dialog management techniques are theoretically well-grounded and appear able to deliver these capabilities. That said, several important research challenges remain, and chief among these is scalability: tracking many hypotheses for the current dialog state is computationally difficult, and performing planning using this representation is exceptionally difficult. Even so, early results show that approximate techniques appear to reduce the complexity dramatically with little loss of performance [34.50]. There is more work to do, but advanced dialog management techniques should begin to filter into mainstream commercial applications in the next few years.

34.4.4 Speech Mining

As the volume of accumulated speech data grows, we are faced with the analytical challenge of finding information of interest for the researchers, developers, executives, and customers, that can explain what is happening, and why, within a deployed service. Mining heterogeneous spoken dialog data for the purpose of improving system operation and extracting business intelligence is a fertile growth area for new research initiatives at many research and industrial labs around the world.

Data mining is concerned with the science, technology, and engineering of discovering patterns and extracting potentially useful or interesting information automatically or semiautomatically from data [34.51]. Data mining was introduced in the 1990s and has deep roots in the fields of statistics, artificial intelligence, and machine learning. With the advent of inexpensive storage space and faster processing over the past decade or so, data mining research has started to penetrate new grounds in areas of speech and audio processing as well as spoken language dialog. This has been fueled by the influx of audio data that are becoming more widely available from a variety of multimedia sources including webcasts, conversations, music, meetings, voice messages, lectures, television, and radio. Algorithmic advances in automatic speech recognition have also been a major, enabling technology behind the growth in data mining. Current state-of-the-art large-vocabulary, continuous speech recognizers are now trained on a record amount of data – several hundreds of millions of words and thousands of hours of speech. Pioneering research in robust speech processing, large-scale discriminative training, finite-state automata, and statistical hidden Markov modeling have resulted in

real-time recognizers that are able to transcribe spontaneous speech with a word accuracy exceeding 85%. With this level of accuracy, the technology is now highly attractive for a variety of speech mining applications.

Speech mining research includes many ways of applying machine learning, speech processing, and language processing algorithms to benefit and serve commercial applications. It also raises and addresses several new and interesting fundamental research challenges in areas of prediction, search, explanation, learning, and language understanding. These basic challenges are becoming increasingly important in revolutionizing business processes by providing essential sales and marketing information about services, customers, and product offerings. They are also enabling a new class of learning systems to be created that can infer knowledge and trends automatically from data, analyze and report application performance, and adapt and improve over time with minimal or zero human involvement.

Effective techniques for mining speech, audio, and dialog data can impact numerous business and government applications. The technology for monitoring conversational speech to discover patterns, capture useful trends, and generate alarms is essential for intelligence and law-enforcement organizations as well as for enhancing call centers operations. It is useful for analyzing, monitoring, and tracking customer preferences and interactions to better establish customized sales and technical support strategies. It is also an essential tool in media content management for searching through large volumes of audio warehouses to find information, documents, and news.

34.4.5 Mobile Devices

The mobile devices market continues to expand. While cell phones are predicted to reach steady-state volumes of about a billion devices a year starting in 2009, the sophistication and variety of additional services continually stresses the existing user interface. Many services remain undiscovered by the majority of cell phone users.

Manufacturers continue to produce specialized devices. The IPod music player by Apple has been a tremendous success, as has been the Razor cell phone by Motorola. In both these cases, the device capabilities are secondary to their industrial design. The prediction for the future is that the form of these devices and their user interfaces have become more important than the actual function of these devices. Consumer demand, at least in the United States, seems to be driven by marketing and consumer appeal.

That said, there are new markets opening because of recent advances. The government is exploring handheld electronic platforms for the soldier of the future. Handheld devices are being used in advanced medical facilities for patient information on demand, and for prescription ordering directly from the doctor to the pharmacy. Handheld navigation systems are a rapidly growing market segment, as larger memories, high-resolution designs, and accurate GPS location systems offer outstanding performance value.

All of these devices could incorporate speech interfaces. In navigation systems, speech output is essential to allow hands-free/eyes-free driver directions. Speech recognition is clearly the input method of the future, but the exact course of the introduction of this interface technology remains unclear.

34.5 Conclusion

In this chapter, we have presented a broad view of speech technologies from the viewpoint of the users of the various technologies. We showed how, beginning in the 1980s, even though speech recognition technologies were not very advanced, there were many *lower-hanging-fruit* applications that were deployed and that were tremendously successful. These applications, although quite simple (i. e., command-and-control applications), nonetheless provided real value to the people and/or businesses that deployed them and were the basis of continued research throughout the private and public sectors. Moving into the 1990s, speech services and speech engine products moved into the mainstream of network-based IVR services. Later in the decade and through the first decade of the 2000s, increased innovations and advances in microprocessor technologies enabled speech technologies to be a mainstay on small devices, such as cell phones, and enabled a new class of complex network-based speech services that begin to understand human speech instead of merely recognizing the words, thus enabling more-human-like interactions between people and machines.

Where is the industry heading? We will see much more automation of telecommunications services, and

for individuals, access to information and services at any time, with any device, and from anywhere, as well as in any language will be the norm. New advances in speech technologies will allow people, governments, educational institutions, and businesses to be informed immediately of any important event and to solve problems automatically. In this networked economy of *anytime, anywhere, and any channel communications*, customers continue to demand services that are quick, convenient, comprehensive, and personalized – this is, where speech is a necessary and valuable innovation. Business is done in real time and conducted with high mobility. Call centers and government services must evolve into 24×365 multimedia-based service centers that combine networking with voice interaction, web technologies, and natural-language voice processing. Speech researchers, though advancing the state of art from an academic exercise to over a billion-dollar market in about 30 years, have still only scratched the surface of the algorithms and capabilities that are required for speech to become ubiquitous in all facets of our lives. This is the challenge we all face, and the fun we will all have as we move forward to make this *future* become a reality.

References

34.1 J.R. Pierce: Whither speech recognition?, J. Acoust. Soc. Am. **46**(4), 1029–1051 (1969)

34.2 K.H. Davis, R. Biddulph, S. Balashek: Automatic recognition of spoken digits. In: *Communication Theory*, ed. by W. Jackson (Butterworths, London 1953)

34.3 A. Lolje, M. Riley, D. Hindle, F. Pereira: The AT&T 60000 word speech-to-text system, Proc. Spoken Language Technology Workshop (Morgan Kaufmann, Austin 1995) pp. 162–165

34.4 L. Rabiner, B.-H. Juang: *Fundamentals of Speech Recognition* (Prentice Hall, Englewood Cliffs 1993)

34.5 F.C. Pereira, M. Riley: Speech recognition by composition of weighted finite automata. In: *Finite-State Devices for Natural Language Processing*, ed. by E. Roche, Y. Schabes (MIT Press, Cambridge 1997)

34.6 V. Goffin, C. Allauzen, E. Bocchieri, D. Hakkani-Tur, A. Ljolje, S. Parthasarathy, M. Rahim, G. Riccardi, M. Saraclar: The AT&T watson speech recognizer, Proc. IEEE ICASSP (2005)

34.7 J. Huang, B. Kingsbury, L. Mangu, M. Padmanabhan, G. Saon, G. Zweig: Recent improvements in speech recognition performance on large conversational speech, Proc. ICSLP (2000)

34.8 B. Atal: Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification, J. Acoust. Soc. Am. **55**(6), 1304–1312 (1974)

34.9 K. Vintsyuk: Speech discrimination by dynamic programming, Kibernetika **4**, 81–88 (1968)

34.10 F. Jelinek: Continuous speech recognition by statistical methods, Proc. IEEE **64**(4), 532–556 (1976)

34.11 L.R. Rabiner: A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE **77**(2), 257–286 (1989)

34.12 F. Jelinek: *Statistical Methods for Speech Recognition* (MIT Press, Cambridge 1997)

34.13 L.R. Bahl, P.F. Brown, P.V. De Souza, R.L. Mercer: Maximum mutual information estimation of HMM parameters for speech recognition, Proc. IEEE ICASSP (1986)

34.14 M.H. Cohen, J.P. Giangola, J. Balogh: *Voice User Interface Design* (Addison Wesley, Boston 2004)

34.15 A. Smola, P. Bartlett, B. Scholkopf, D. Schuurmans: *Advances in Large Margin Classifiers* (MIT Press, Cambridge 2000)

34.16 R. Schapire, M. Rochery, M. Rahim, N. Gupta: Incorporating prior knowledge into boosting, Proc. Nineteenth Int. Conf. Machine Learning (2002)

34.17 J. Baker: The Dragon system – an overview, IEEE Trans. ASSP **23**(1), 24–29 (1975)

34.18 A. Gorin, G. Riccardi, J. Wright: How May I Help You?, Speech Commun. **23**, 113–127 (1997)

34.19 http://www.nexidia.com

34.20 http://www.verint.com

34.21 R. Natarajan, B. Prasad, B. Suhm, D. McCarthy: Speech enabled natural language call routing: BBN call director, Proc. Int. Conf. Spoken Language Process. (2002)

34.22 L. Lee, R. Rose: A Frequency Warping Approach to Speaker Normalization, IEEE Trans. Speech Audio Process. **6**, 49–60 (1998)

34.23 D.A. Reynolds, R.C. Rose: Robust text-independent speaker identification using Gaussian mixture speaker models, IEEE Trans. Speech Audio Process. **3**(1), 72–83 (1995)

34.24 X.D. Huang, A. Acero, H.-W. Hon: *Spoken Language Processing* (Prentice Hall, Englewood Cliffs 2001)

34.25 M. Rahim, B.-H. Juang: Signal bias removal by maximum likelihood estimation for robust speech recognition, IEEE Trans. Speech Audio Process. **4**(1), 19–30 (1996)

34.26 S. Bangalore, G. Riccardi: Stochastic finite-state models for spoken language machine translation, Mach. Transl. **17**(3), 165–184 (2002)

34.27 N. Gupta, G. Tur, D. Hakkani-Tür, S. Bangalore, G. Riccardi, M. Rahim: The AT&T spoken language understanding system, IEEE Trans. Audio Speech Lang. Process. **14**(1), 213–222 (2006)

34.28 G. Riccardi, D. Hakkani-Tür: Active and unsupervised learning for automatic speech recognition, Proc. 8th European Conf. Speech Commun. and Technol. (2003)

34.29 S. McGlashan: Voice Extensible Markup Language (VoiceXML) Version 2.0 (2004) (http://www.w3.org/TR/2004/PR-voicexml20-20040203)

34.30 R. Nakatsu: Anser – An application of speech technology to the Japanese banking industry, Computer **23**(8), 43–48 (1990)

34.31 http://www.nuance.com

34.32 http://www.tellme.com

34.33 http://www.bevocal.com

34.34 http://www.telureka.com

34.35 http://www.convergys.com

34.36 http://www.west.com

34.37 J. Wilpon, L.R. Rabiner, C.H. Lee, E.R. Goldman: Automatic recognition of keywords in unconstrained speech using hidden Markov models, IEEE Trans. Acoust. Speech Signal Process. **38**(11), 1870–1878 (1990)

34.38 W.T. Hartwell, M.A. Johnson, J. Picone: Automatic speech recognition using echo cancellation, US Patent **4,914,692** (1990)

34.39 V. Franco: Automation of operator services at AT&T, Proc. Voice (1993)

34.40 S. Shanmugham, D. Burnett: Media Resource Control Protocol Version 2 (MRCPv2) (http://tools.ietf.org/wg/speechsc/draft-ietf-speechsc-mrcpv2/draft-ietf-speechsc-mrcpv2-09.txt)

34.41 http://www.w3.org/TR/xhtml+voice

34.42 L.R. Rabiner: Applications of voice processing to telecommunications, Proc. IEEE **82**(2), 199–228 (1994)

34.43 H. Sakoe, C. Chiba: Dynamic programming algorithm optimization for spoken word recognition, IEEE Trans. Acoust. Speech Signal Process. **ASSP-26**, 43–49 (1978)

34.44 J. Cooperstock: From the flashing 12:00 to a usable machine: Applying UbiComp to the VCR (http://acm.org/sigchi/chi97/proceedings/short-talk/jrc.htm)

34.45 A.H. Gray Jr., J.D. Markel: Distance measures for speech processing, IEEE Trans. ASSP **24**(5), 380–391 (1976)

34.46 M. Przybocki, A. Martin: *NIST's Assessment of Text Independent Speaker Recognition Performance* (2005)

34.47 M. Johnston, S. Bangalore, G. Vasireddy, A. Stent, P. Ehlen, M. Walker, S. Whittaker, P. Maloor: MATCH: An architecture for multimodal dialogue systems, Proc. 40th Annual Meeting of the Association for Computational Linguistics (2002)

34.48 http://www.saltforum.org/saltforum/downloads/SALT1.0.pdf

34.49 T. Paek, E. Horvitz: Conversation as action under uncertainty, Proc. Conf. Uncertainty in Artificial Intelligence (UAI) (2000)

34.50 J.D. Williams: *Partially Observable Markov Decision processes for Spoken Dialog Management*, Ph.D. Thesis (University of Cambridge, Cambridge 2006)

34.51 I. Witten, E. Frank: *Data Mining* (Morgan Kaufmann, Austin 1999)

30. Towards Superhuman Speech Recognition

M. Picheny, D. Nahamoo

After over 40 years of research, human speech recognition performance still substantially outstrips machine performance. Although enormous progress has been made, the ultimate goal of achieving or exceeding human performance – *superhuman* speech recognition – eludes us. On a more-prosaic level, many industrial concerns have been trying to make a go of various speech recognition businesses for many years, yet there is no clear *killer app* for speech. If the technology were as reliable as human perception, would such *killer apps* emerge?

Either way, there would be enormous value in producing a recognizer with superhuman capabilities. This chapter describes an ongoing research program at IBM that attempts to address achieving superhuman speech recognition performance in the context of the metric of word error rate. First, a multidomain conversational test set to drive the research program is described. Then, a series of human listening experiments and speech recognition experiments based on the test set is presented. Large improvements in recognition performance can be achieved through a combination of adaptation, discriminative training, a combination of knowledge sources, and simple addition of more data. Unfortunately, devising a set of informative listening tests synchronized with the multidomain test set proved to be more difficult than expected because of the highly informal nature of the underlying speech. The problems encountered in performing the listening tests are presented along with suggestions for future listening tests. The chapter concludes with a set of speculations on the best way for speech recognition research to proceed in the future in this area.

30.1 Current Status

After over 40 years of research, human speech recognition performance still substantially outstrips machine performance. Although enormous progress has been made, the ultimate goal of achieving or exceeding human performance – *superhuman* speech recognition – eludes us. In addition, it is fair to say that there have been no recent *breakthroughs* in the speech recognition area – progress over the last several years, though continual, has been evolutionary rather than revolutionary. Can we achieve levels of superhuman speech recognition in our lifetimes through evolutionary approaches, or do we need to make radical changes to our methodologies?

On a more-prosaic level, many industrial concerns have been trying to make a go of various speech recognition businesses for many years, yet there is no clear *killer app* for speech that would guarantee the huge revenue needed to justify the expense of investing in the technology. Is this because there really is no value in the

technology, or that the technology just needs additional incremental improvements, or is this because the levels of accuracy must reach human performance or higher to be regarded as sufficiently reliable to serve as a user interface?

Either way, there would be enormous value in producing a recognizer with superhuman capabilities. This chapter describes an ongoing research program at IBM that attempts to address some of these questions, focusing on word error rate as a metric. It is recognized that this is only one of many dimensions in which speech recognition performance can be measured, and it is hoped that this chapter will trigger corresponding studies in related areas, such as concept and meaning extraction. First, we describe a multidomain conversational test set established to drive the research program (Sect. 30.2). Then, we describe a series of human listening experiments that attempt to determine the best set of research investments to achieve the goal of superhuman speech recognition (Sect. 30.3), a set of recognition experiments that begin to address methodologies for designing systems to achieve superhuman performance (Sect. 30.4), and a set of speculations on the best way for research to proceed in the future in this area (Sect. 30.5).

30.2 A Multidomain Conversational Test Set

We had a number of goals in mind when we designed the test set. First, the test set had to cover a reasonably broad range of conversational applications and contain data representing key challenges to reliable recognition including various forms of acoustic interference, speech from non-native speakers, and a large recognition vocabulary. Second, the test set had to include at least one component that is readily available to other researchers to facilitate comparisons between our recognizers and those developed externally. Third, the test set needed to be reasonably small, to facilitate rapid turnaround of experiments. For all experiments reported here, we used a test set composed of the following five parts.

swb98: The switchboard portion of the 1998 hub 5e evaluation set [30.1], consisting of 2 h of telephone-bandwidth (8 kHz) audio. The data were collected from two-person conversations between strangers on a preassigned topic. A variety of telephone channels and regional dialects are represented in the data.

mtg: An initial release of the Bmr007 meeting from the ICSI (International Computer Science Institute) meeting corpus [30.2], consisting of 95 minutes of audio. The data were collected from eight speakers wearing either lapel microphones or close-talking headsets. This meeting involved eight speakers: five native speakers of American English (two females and three males), and three non-native speakers (all males). Although the data is wide bandwidth (16 kHz), the primary challenge in this test set is the presence of background speech in many of the utterances. The crosstalk problem is especially severe for speakers recorded using lapel microphones.

cc1: 0.5 h of audio from a call center. The data are collected from customer-service representatives (CSRs) and customers calling with service requests or problems. The primary challenge in this test is acoustic interference: a combination of nonlinear distortion from speech compression, background noise from both the CSR and customer sides, and intermittent beeps on the channel, which are played to remind the customer that the call is being recorded. The data is telephony bandwidth but otherwise relatively quiet.

cc2: 0.5 h of audio from a second call center. The recordings are from a different center than the cc1 test set, but cover similar subject matter and have similar, poor acoustics. This data set has no information associating speakers with sets of utterances, which poses problems for speaker and channel adaptation. The data is telephony bandwidth but otherwise relatively quiet.

vm: Test data from the IBM voicemail corpus, consisting of 1 hour of audio. This material was previously reported on as the E-VM1 test set [30.3], and is a superset of the test data in the voicemail corpus part I and part II distributed by the LDC (Linguistic Data Consortium). Unlike the other tests, the voicemail data are conversational monologues. The acoustic quality of the data is generally quite high, although loud clicks caused by the speaker hanging up at the end of some messages can pose problems for feature normalization – especially

Table 30.1 Characteristics of the multidomain conversational test set used for listening and recognition experiments

Task	Number of hours	Number of segments
swb98	2.0	3500
mtg	1.5	2060
cc1	0.5	978
cc2	0.5	1033
vm	1.0	1033

normalization of c0 based on the maximum value of c0 within an utterance. This test set also has no information associating speakers with sets of utterances. The data is telephony bandwidth but otherwise relatively quiet.

The test sets were segmented into utterances suitable for recognition using a variety of means. The default LDC segmentation was used for the swb98 data. An automatic segmenter was used on the mtg data. The cc1 and cc2 data came in the form of calls and were automatically segmented into smaller units; the vm data came in the form of individual messages and was also automatically segmented into smaller units. A summary of the total number of hours and segments for each test set is given in Table 30.1.

30.3 Listening Experiments

Humans use whatever information is available to aid recognition performance. In a recent classic paper, *Lippmann* [30.4] compared machine and human recognition performance across a wide variety of stimuli. He demonstrated that human performance far exceeded machine performance with minimal linguistic information (digits, letters, nonsense sentences), minimal acoustic information (speech in noise), and various combinations of both (telephone conversations). Table 30.2 shows human versus machine performance for digits, letters, sentences from the *Wall Street Journal*, and telephony conversations. In all four cases, human performance was significantly better than machine performance, sometimes by more than an order of magnitude. Although these numbers were obtained some years ago, one would be hard pressed to argue that more than a factor of two improvement in recognition has occurred in the last 10 years, so the gap is still substantial.

Many studies have suggested that humans can accurately identify words with as little as two seconds of surrounding context [30.5, 6]. *Allen* [30.7] presents strong evidence that humans can highly accurately recognize phonemes without any linguistic context, and also suggests that the enormous robustness of human speech perception across degradations in channel conditions arises from the clever processing of many independent frequency bands in the auditory system. Internal experiments performed at IBM by Jelinek and his associates in the late 1980s suggest that human performance in word prediction from text (the *Shannon Game* [30.8]) is over three times better than the language models then (and unfortunately, still currently) used in speech recognition when humans are presented with full sentential context. No studies seem to exist that assess the relative importance of these information sources, so we initiated a series of tests to try to ascertain the performance of humans on the multidomain test set.

Table 30.2 Human versus machine recognition word error rates across a variety of tasks (after *Lippmann* [30.4])

Task	Machine performance %	Human performance %
Connected digits	0.72	0.009
Letters	5	1.6
Resource management	3.6	0.1
WSJ (Wall Street Journal)	7.2	0.9
Switchboard	43	4

30.3.1 Baseline Listening Tests

The first set of experiments attempted to obtain baseline performance figures on how well humans could actually recognize the multidomain test set. Random segments were chosen from the test set and played to two listeners over headphones. In one experiment, the listeners were allowed to listen only one time to the utterances, and in the second experiment, the listeners were allowed to hear the utterances multiple times. The utterances were randomized across the test set both in terms of domain and in terms of order. In addition, long utterances were segmented into no longer than 2–3 second chunks, as the memory load for utterances that were longer essentially made the single-pass task intractable.

The summary results are shown in Table 30.3. The error rates are much higher than those presented in [30.4] on similar data. In addition, no appreciable reduction in error rate is seen when the listener is allowed to listen multiple times to the test utterance. The error rates

Table 30.3 Summary of the multidomain test set base listening test results

Condition	Word error rate
Listen once	25.6
Listen multiple times	21.5

Table 30.4 Breakdown of base listening test results across corpora (listen-once case)

Corpus	Word error rate
swb98	25.1
mtg	28.1
cc1	19.6
cc2	32.2
vm	18.8

by corpus were broken down to see if there were any obvious dependencies on the domain materials. The breakdown is shown in Table 30.4. As can be seen, there is no clear dependency on any one particular corpus, though it is clear, for example, that the voicemail data is more comprehensible than the call center data, perhaps because the talkers know that they are leaving a message that is to be listened to after the fact.

Why is there such a discrepancy in the results here and the results reported in [30.4]? There are a number of possible causes. First, the earlier experiments were performed on longer segments of speech, listened to multiple times by each listener. Second, the earlier experiments may not have randomized the segments across speaker. Therefore, the listener may have been able to take advantage of both task adaptation and speaker/channel adaptation in the earlier results. In an attempt to try to tease these issues apart, we embarked upon a more-ambitious series of listening tests, described in the next subsection.

30.3.2 Listening Tests to Determine Knowledge Source Contributions

The processes that determine how humans recognize speech are still subjects of ongoing research and will remain so for some time to come. The goal of a speech recognition professional is not so much to understand how humans recognize speech but to try to utilize human strategies of speech recognition as a guide to improve

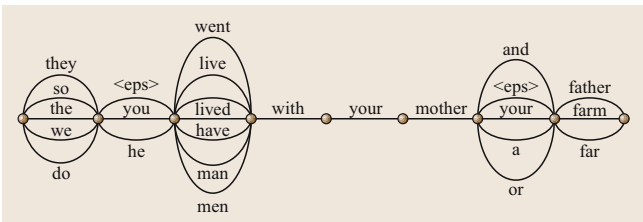


Fig. 30.1 A typical word confusion network, also known as a *sausage*

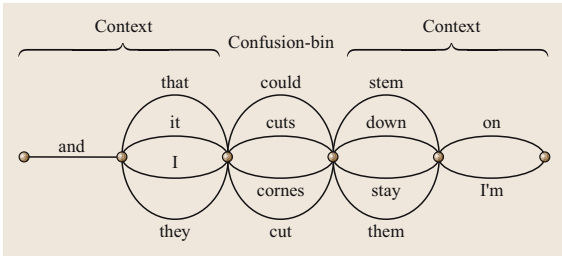


Fig. 30.2 Word recognition test for the *listening* condition depicting only the audio information and sausage structure presented to the subjects

the performance of today's speech recognition systems. As such, the next set of listening experiments attempt to determine what sources of knowledge humans can use in the context of performance of today's speech recognition systems.

Currently, one of the most interesting aspects of speech recognition systems is that even when they produce relatively high word error rates (e.g., 30%), it is possible for them to produce extremely compact representations of the search space in which the best possible, or oracle, error rate is very low (e.g., 10%). These representations take the form of a confusion network (typically referred to as a *sausage*), as illustrated in Fig. 30.1. In this example, the correct word sequence is *so you lived with your mother and father,* although *do you live with your mother and father* and many other sequences are also possibilities. (The word that was ranked highest by the recognizer appears at the top of each segment.)

In this set of listening experiments, sausage structures produced by a state-of-the-art recognizer were presented to human subjects, and they were asked to select the best word sequence either with accompanying audio (*listening*) or with accompanying long language model context (*comprehension*). The hope was to assess the relative contributions of acoustic and language information as possible information sources on top of a pre-existing speech recognition system.

More specifically, in the listening condition (Fig. 30.2) subjects were presented with audio containing two words to the left, the target word and two words to the right of the word target, and asked to identify the target center word. In the comprehension condition (Fig. 30.3), the subject was given five segments of text history and asked to determine the best path through the sausage network without corresponding audio. In these experiments, we used audio data from the RT03 evaluation data set from LDC [30.9] and from the

Table 30.5 Word and oracle recognition error rates for listening tests to determine relative knowledge source contributions

Corpus	Word error rate	Oracle error rate
RT03	29.2	8.0
MALACH	28.3	9.5

Table 30.6 Listener error rates when presented with audio context in the form of the surrounding words (listening) and word context in the form of sausages (comprehension)

Corpus	Base recognition word error rate	Listening	Comprehension
RT03	29.2	24.2	30.6
MALACH	28.3	27.3	32.0

MALACH corpus [30.10]. A summary of the word error rates and oracle error rates for the sausages are shown in Table 30.5. As can be seen, the oracle error rates are much lower than the word error rates, implying that, if there is useful information that humans can utilize in the surrounding context, error rates can be made to drop significantly.

Sausages as described above were presented to a set of 20 listeners in random order with respect to speaker, channel, and topic. Each listener saw or heard 100 sausages either under the listening or comprehension conditions. The results are shown in Table 30.6. As can be seen from the results, neither source of information dramatically improved the listener's ability to choose the correct path through the sausage data. The presence of acoustic information helped somewhat more than textual context, but not in some definitive sense. Subjectively speaking, the task was surprisingly difficult to perform – the conversations had significant sections of quite unintelligible murmuring which may not be critical for comprehension but when segments are selected completely at random may significantly affect the overall error rate. In retrospect, it might have been useful to include a condition in which both long-span acoustic and language model information was included to ensure that humans could perform the task with *infinite* context, and also have a contrastive condition in which the target speaker is fixed across a variety of utterances. There were issues associated with occasional misalignments of the target word text and the underlying audio and some segmentation artifacts, but these did not occur frequently enough to affect the error rates in a significant fashion.

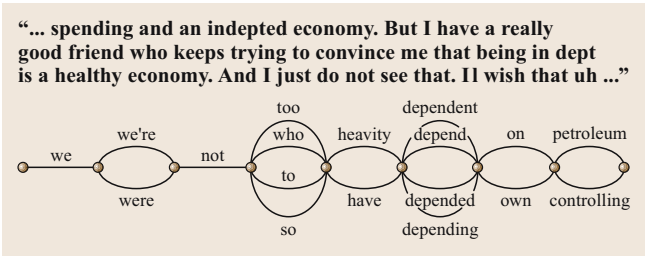


Fig. 30.3 Word recognition test for the comprehension condition depicting only the text information and the sausage structure presented to the subjects

What these initial experiments illustrate are some of the difficulties in performing human listening experiments as a guide for a technical research agenda in speech recognition. Unfortunately, we were unable to adequately tease out where research efforts should be focused to achieve the maximum benefit, and were left to rely on our technical intuition as to how best to proceed.

30.4 Recognition Experiments

To achieve superhuman speech recognition, it is useful to decompose speech recognition into a set of semi-independent tasks. This is shown in Table 30.7. The error rates are rough guides and should not be taken

Table 30.7 Recognition performance in terms of word error rates for different styles and types of speech

Task	Speech style	Target	Channel	Word error rate
Dictation	Well formed	Computer	Full BW (bandwidth)	< 4 [30.11]
Broadcast news	Usually well formed	Spontaneous	Audience	8.6 [30.12]
DARPA communicator	Spontaneous	Computer	Telephone BW	15.1 [30.13]
SWB	Spontaneous	Person	Telephone BW	15.2 [30.14]
Voicemail	Spontaneous	Person	Telephone BW	27.9 [30.3]
Meetings	Spontaneous	People	Far-field	50 [30.15]

literally. As can be seen, the main determinants of recognition difficulty are the mode of speech (is one talking to a machine or to a person), and the channel/environment. Needless to say, since people obviously communicate in such modes and across such environments quite freely, a superhuman speech recognizer at a minimum must be robust across these dimensions.

30.4.1 Preliminary Recognition Results

In an attempt to get a feeling for the domain sensitivity of current speech recognition systems, a set of experiments was performed. Three corpora were available for training – transaction data [commands, digits, Defense Advanced Research Projects Agency (DARPA) communicator data], switchboard data, and voicemail data. Four corpora were available as test sets – continuous digits, names, switchboard test data (swb98, above) and voicemail test data (vm, above). Systems were trained individually on these corpora. In addition, a common technique – multistyle training – was employed to produce systems in which the training data from the switchboard and voicemail were combined, and one in which all three training corpora were combined. The training and decoding were done as described in Sect. 30.4.2. In all cases, for decoding, a language model suitable for the test data was utilized.

The results are shown in Table 30.8. As can be seen, significant degradation can result when data from one mode of speaking is decoded with models trained on a different mode of speaking. Note also that multistyle training across different corpora to some extent reduces the cross-corpus training degradation effects, but does not always allow one to completely recover the performance levels obtained when test and training data are matched to each other. This implies that blind combination of a huge amount of data from multiple sources is not likely to allow us to reach levels of superhuman performance in cross-domain experiments (and the best way in which to combine language model data has not yet been addressed).

30.4.2 Results on the Multidomain Test Set

A conscious decision was made to focus on telephony transcription data as an initial challenge. Therefore, we continued to utilize the multidomain test corpus described in Sect. 30.2. To deal with the broad range of material present in the multidomain test set, we employed a recognition strategy based on multiple passes of recognition interleaved with unsupervised acoustic model adaptation and on a combination of recognition hypotheses from systems using disparate feature sets and acoustic models. We first describe the basic techniques used in the benchmark system for signal processing, acoustic modeling, adaptation, and language modeling, then we describe the architecture of the recognition system and present the performance of the system on the multidomain test corpus at various stages of processing.

Signal Processing

The systems in this work use either mel-frequency cepstral coefficients (MFCC) or perceptual linear prediction (PLP) features as raw features. The MFCC features are based on a bank of 24 mel filters spanning 0–4.0 kHz. The PLP features are based on a bank of 18 mel filters spanning 0.125–3.8 kHz and use a 12th-order autoregressive analysis to model the auditory spectrum. Both feature sets are based on an initial spectral analysis that uses 25 ms frames smoothed with a Hamming window, a 10 ms frame step, and adds the equivalent of 1 bit of noise to the power spectra as a form of flooring. Both feature sets also are computed using periodogram averaging to reduce the variance of the spectral estimates. The final recognition feature set for all systems are generated by concatenating raw features from nine consecutive frames and projecting to a 60-dimensional (60-D) feature space. The projection is a composition of a discriminant projection (either linear discriminant analysis or heteroscedastic discriminant analysis [30.16]) and a diagonalizing transform [30.17, 18].

Prior to the projection to the final, 60-D recognition feature space, the raw features are normalized. Three

Table 30.8 Preliminary test results. The training corpora are listed across the top and the test corpora are represented by the rows

	Training corpus				
	Transactions	SWB	Voicemail	Voicemail + SWB	All
Names	4.4	6.4	8.6		5.3
Digits	1.3	1.9	2.4		1.4
SWB		39	57	46	
Voicemail		47	36	37	

different normalization schemes are used by different systems in this work:

1. utterance-based mean normalization of all features;
2. utterance-based mean normalization of all features except c0 and maximal normalization of c0; and
3. side-based mean and variance normalization of all features except c0 and maximal normalization of c0. In maximal normalization of c0, the maximum value of c0 within an utterance is subtracted from c0 for all frames in the utterance. The estimate of variance is based solely on frames for which c0 exceeds a threshold with respect to the maximum value of c0 in the utterance. This is intended to ensure that the variance is only computed from speech frames.

Acoustic Modeling

We use an alphabet of 45 phones to represent words in the lexicon. Each phone is modeled as a three-state, left-to-right hidden Markov model (HMM). Acoustic variants of the HMM states are identified using decision trees that ask questions about the surrounding phones within an 11-phone context window (±5 phones around the current one). Systems may employ *word-internal* context, in which variants are conditioned only on phones within the current word, or *left* context, in which variants are conditioned on phones within the current and the preceding words.

The majority of the systems described in this work model the leaves of the phonetic decision trees using mixtures of diagonal-covariance Gaussian distributions that are trained using maximum-likelihood estimation (MLE). Subject to a constraint on the maximum number of Gaussians assigned to a leaf, the number of mixture components used to model a leaf is chosen to maximize the Bayesian information criterion (BIC),

$$F(\theta) = \log P(X_s|s,\theta) - \frac{\lambda}{2}|\theta|\log(N_s)\,, \qquad (30.1)$$

where $P(X_s|s,\theta)$ is the total likelihood of the data points X_s that align to leaf s under model θ, N_s is the number of such points, and $|\theta|$ is the total number of parameters in model θ. The overall size of an acoustic model may be adjusted by changing the weight on the BIC penalty term, λ. The acoustic models for all recognizers are trained on 247 h of switchboard data and 18 hours of Callhome English data. Early experiments revealed that, after acoustic adaptation, no benefit was obtained by combining other sources of data (such as voicemail data, as described in Sect. 30.4.1) so no additional data was combined with the SWB (switchboard) data.

Two systems described in this work employ alternative acoustic models. One system models leaves using mixtures of diagonal-covariance Gaussian distributions that are discriminatively trained using maximum mutual information estimation (MMIE). In our MMIE training, we collect counts by running the forward–backward algorithm on a statically compiled decoding graph, using beam pruning to constrain the size of the search space [30.19]. This lets us exploit technology developed for fast decoding of conversational speech [30.20] for fast MMIE training as well. The second system models leaves with subspace precision and means (SPAM) models [30.21, 22]. SPAM models provide a framework for interpolating between diagonal-covariance and full-covariance Gaussian mixture models in terms of model complexity and model accuracy. Unlike the diagonal-covariance Gaussian models in this work, SPAM models do not directly use BIC-based model selection.

Canonical Acoustic Models

We use two feature-space transformations, vocal-tract-length normalization (VTLN) [30.23] and maximum-likelihood feature-space regression (FMLLR) [30.24], in an adaptive training framework to train *canonical* acoustic models. The goal of canonical training is to reduce variability in the training data due to speaker- and channel-specific factors, thereby focusing the acoustic model on variability related to linguistic factors. At test time, the feature-space transforms are estimated in an unsupervised fashion, using results from earlier decoding passes.

Our implementation of VTLN uses a set of 21 warp factors that cover a ±20% linear rescaling of the frequency axis. The VTLN frequency warping is applied prior to mel binning in the feature computation. The VTLN warp factor for a speaker is chosen to maximize the likelihood of frames that align to vowels and semivowels under a voicing model that uses a single, full-covariance Gaussian per context-dependent state. Approximate Jacobian compensation of the likelihoods is performed by adding the log determinant of the sum of the outer products of the warped cepstra to the average frame log likelihood.

The FMLLR transformation is an affine transformation of the features in the final, 60-D recognition feature space that maximizes the likelihood of a speaker's data under an acoustic model. FMLLR is equivalent to constrained maximum-likelihood linear regression (MLLR) [30.24], where the MLLR transform is applied to both the means and covariances of the acoustic model. In the remainder of the paper, we will refer to canon-

ical models that use VTLN features as VTLN models and to canonical models that use VTLN features and an FMLLR transformation as SAT (speaker-adaptive trained) models.

Acoustic Model Adaptation

At test time, we also use MLLR adaptation [30.25] of model means to further adapt the recognition system to the specific speaker and environment. Systems that use diagonal Gaussian mixture acoustic models perform two rounds of MLLR. The first round estimates one MLLR transform for all speech models and one MLLR transform for all nonspeech models, and new recognition hypotheses are generated with the adapted models. In the second round, multiple MLLR transforms are estimated using a regression tree and a count threshold of 5000 to create a transform for a regression class. The system using SPAM models performs a single round of adaptation in which a single MLLR transform for all models and a new FMLLR transform are estimated.

Language Modeling and Recognition Lexicon Design

The data used to train the language models consist of 3 million switchboard words, 16 million broadcast news words, 1 million voicemail words and 600,000 call center words. For the initial rescoring of the word internal lattices we used a four-way interpolated language model, each of the components being a back-off 3-gram LM (language model) using modified Kneser–Ney smoothing [30.26]. The mixture weights ($0.45 \cdot \mathrm{Swb} + 0.25 \cdot \mathrm{BN} + 0.15 \cdot \mathrm{VM} + 0.2 \cdot \mathrm{CC}$) are optimized on a held-out set containing 5% of each of the training corpora. For the final rescoring of the left-context lattices the 3-gram mixture components are replaced with 4-gram language models, keeping the mixture weights the same. The 34,000-word vocabulary used in our experiments consists of all the high-count words from our training corpora. The pronunciation dictionary consists of 37,000 entries, yielding a ratio of 1.09 pronunciations per word in the vocabulary. Table 30.9 shows the perplexities and the out-of-vocabulary (OOV) rates for each of the five test sets.

Table 30.9 Perplexities and OOV rates across different test sets

Test set	Perplexity 3gm LM	Perplexity 4gm LM	OOV rate (%)
swb98	94.16	90.08	0.3
mtg	146.45	142.58	0.7
cc1	111.69	106.42	0.3
cc2	52.95	49.30	0.1
vm	94.66	89.32	1.1

Recognition Process and Performance

Recognition of data system proceeded as follows:

P1 Speaker-independent decoding. The system uses mean-normalized MFCC features and an acoustic model consisting of 4078 left context-dependent states and 171 000 mixture components. Decoding is performed using IBM's rank-based stack decoding technology [30.27].

P2 VTLN decoding. VTLN warp factors are estimated for each speaker using forced alignments of the data to the recognition hypotheses from P1, then recognition is performed with a VTLN system that uses mean-normalized PLP features and an acoustic model consisting of 4440 left context-dependent states and 163 000 mixture components. Decoding is performed using IBM's rank-based stack decoder. In the cc2 and vm test sets, which have no speaker information, VTLN warp factors are estimated for individual utterances.

P3 Lattice generation. Initial word lattices are generated with a SAT system that uses mean-normalized PLP features and an acoustic model consisting of 3688 word-internal context-dependent states and 151 000 mixture components. FMLLR transforms are computed using recognition hypotheses from P2. The lattices are generated with a Viterbi decoder. The lattices are then expanded to trigram context, rescored with a trigram language model and pruned. In the cc2 and vm test sets, which have no speaker information, FMLLR transforms are estimated for individual utterances.

P4 Acoustic rescoring with large SAT models. The lattices from P3 are rescored with five different SAT acoustic models and pruned. The acoustic models are as follows:

- [A] An MMIE PLP system consisting of 10 437 left context-dependent states and 623 000 mixture components. This system uses maximum-normalization of c0 and side-based mean and variance normalization of all other raw features.
- [B] An MLE PLP system identical to the system of P4A, except for the use of MLE training of the acoustic model.
- [C] An MLE PLP system consisting of 10 450 left context-dependent states and 589 000 mixture components. This system uses mean normalization of all raw features.

Table 30.10 Word error rates (percentage) for the components of the multidomain test set and the overall, average error rate for the corpus. For passes where multiple systems are used (P4–6), the best error rate for a test component is highlighted

Pass	swb98	mtg	cc1	cc2	vm	All
P1	42.5	62.2	67.8	47.6	35.4	51.1
P2	38.7	53.7	56.9	44.1	31.7	45.0
P3	36.0	44.6	46.6	40.1	28.0	39.1
P4A	**31.5**	**39.4**	41.7	38.2	26.7	35.5
P4B	32.3	40.0	**41.3**	39.0	26.7	35.9
P4C	32.5	40.2	42.1	39.9	27.0	36.3
P4D	31.7	40.3	42.6	**37.6**	**25.8**	35.6
P4E	33.0	40.5	43.4	38.8	26.9	36.5
P5A	30.9	**38.3**	**39.4**	36.9	26.1	34.3
P5B	31.5	38.5	**39.4**	37.0	26.5	34.6
P5C	31.6	38.7	41.0	39.4	26.8	35.5
P5D	**30.8**	39.0	41.1	**36.7**	**25.6**	34.6
P5E	32.1	38.9	41.8	36.8	26.4	35.2
P6A	**30.4**	**38.0**	**38.9**	36.5	25.7	33.9
P6B	31.0	38.3	**38.9**	36.4	25.8	34.1
P6C	31.2	38.4	40.1	38.9	26.3	35.0
P6D	**30.4**	38.6	40.8	36.3	**25.5**	34.3
P6E	31.5	38.5	41.6	**35.9**	25.7	34.6
P7	29.0	35.0	37.9	33.6	24.5	32.0

- [D] A SPAM MFCC system consisting of 10 133 left context-dependent states and 217 000 mixture components. The SPAM models use a 120-dimensional basis for the precision matrices. This system uses mean normalization of all raw features.
- [E] An MLE MFCC system consisting of 10 441 left context-dependent states and 600 000 mixture components. This system uses maximum-normalization of c0 and mean normalization of all other raw features.

The FMLLR transforms for each of the five acoustic models are computed from the one-best hypotheses in the lattices from P3. FMLLR transforms are estimated for individual utterances in the vm test set, but on the cc2 test set a single FMLLR transform is estimated from all utterances. The vm test set contains many long utterances [30.28], and the FMLLR estimation procedure has sufficient data, even with very large acoustic models. We found that the cc2 test set contained only very short utterances, and the FMLLR procedure failed to converge on many utterances with the large acoustic models.

P5 Acoustic model adaptation. Each of the five acoustic models are adapted using one-best hypotheses from their respective lattices generated in P4; no cross-system adaptation is performed. As described above, the systems using Gaussian mixture acoustic models are adapted using two sets of MLLR transforms, while the SPAM acoustic model is adapted using an FMLLR transform and an MLLR transform. The lattices from P3 are rescored using the adapted acoustic models and pruned. As in P4, transforms are estimated for individual utterances in the vm test set, but are estimated globally for the cc2 test set.

P6 4-gram rescoring. Each of the five sets of lattices from P5 are rescored and pruned using a 4-gram language model.

P7 Confusion network combination. Each of the five sets of lattices from P6 are processed to generate confusion networks [30.29], then a final recognition hypothesis is generated by combining the confusion networks for each utterance.

The performance of the various recognition passes on the test set is summarized in Table 30.10.

Conclusions

Reasonable recognition performance can be obtained on a broad sample of conversational American English tasks using acoustic models trained only on switchboard

and Callhome data. The results on the mtg set illustrate this point most strongly, for neither the acoustic models nor the language models are trained on meeting data. This supports the observation that the switchboard corpus is representative of the acoustic–phonetic and stylistic properties of conversational American English [30.30].

Multipass decoding with unsupervised adaptation and a combination of disparate systems are effective techniques for achieving good recognition performance on diverse data sources. On this test set, they can reduce the overall error rate from 51.1 to 32.0%.

While system combination can provide consistent gains in recognition performance, they are relatively small for the rather substantial amount of computation incurred. Had we used only the MMIE PLP system and performed consensus decoding instead of the confusion network combination in P7, the overall error rate on the test would have increased to 33.1%. This rather intensive amount of computation is a clear impediment to research and resulted in a major redesign of the system as described in the next section.

30.4.3 System Redesign

As described above, the main problem with the above system was its high level of complexity. Too many passes over too many systems are required – the above system ran in about 5000×RT (real-time), which is really outrageous. If one looks at the culprits in the computation, it is primarily the need to generate many sequences of lattices inefficiently because of an inability to handle all the recognition sources of knowledge in a single pass. For this reason a major redesign of the recognition system was performed to allow for all the knowledge sources needed for recognition to be compiled into a single static network, and many of the multiple passes used to determine warping factors and FMLLR transforms were combined into a single pass or eliminated altogether [30.20]. It was found that this combination not only sped up decoding enormously but also resulted in much better accuracy. Essentially, each pass of lattice generation and pruning was introducing search errors that were eliminated by migrating to a single-pass decoding process. Another innovation that was added was incorporation of a dynamic language model that could automatically modify the interpolation weights in a two-pass recognition process. Each utterance was decoded with a general language model and a lattice was produced. The interpolation weights of the LM components were then adjusted to minimize perplexity of the decoded utterance text, and the new LM that resulted was used to rescore the lattice to produce the final recognition result.

Table 30.11 Recognition results on the multidomain test set after system redesign

Corpus	Previous	After redesign
swb	29.0	27.0
mtg	35.0	33.6
cc1	33.6	24.7
cc2	37.9	36.9
vm	24.5	20.9
Average	32.0	28.6
Number of recognizers	5	1
Speed	5000×RT	20×RT

The results are shown in Table 30.11. As can be seen, not only has the recognition accuracy significantly improved, but the amount of computation dramatically has also decreased. The need for multiple recognizers is essentially eliminated and even the speed of a single recognition pass significantly dropped. Analysis indicated that approximately 2% absolute of the performance gain was attributable to the presence of the adaptive language model.

30.4.4 Coda

After the above set of experiments were completed, a shift in emphasis to focus on public data, such as those present in the DARPA EARS (effective, affordable, reusable speech-to-text) and GALE (global autonomous language exploitation) programs, took place. The 2004 EARS system [30.14] was basically an extension of the system described in Sect. 30.4.3 to three systems (speaker independent, speaker adaptive with diagonal covariances, and speaker adaptive with full covariances), all trained with MPE (minimum phone error) [30.31] and fMPE [30.32]. In addition, significantly more data (2000 h) were used to train the models. Unfortunately, as described in Sect. 30.4.3 the signal processing needed to process the superhuman test set was a modified version of the EARS system signal processing to improve robustness, so it was only possible to decode the swb data component with the EARS system. The result was a 19% word error rate compared to a 27% word error rate on the swb component of the system described in Sect. 30.4.3. Approximately half of the improvement could be attributed to increased data and the rest divided across the new acoustic modeling (MPE and fMPE) and

the use of multiple systems. Assuming that at least some of these improvements would have held up across the other data components, it is safe to say we would have been looking at error rates between 20% and 25% on the overall superhuman corpus. Given the results of the human listening tests (Sect. 30.3.2), at least on short segments of speech, one could almost say that for telephony speech transcription and limited acoustic context, superhuman performance no longer seems completely out of reach.

30.5 Speculation

Although the above sections suggest some progress has been made in the direction of achieving superhuman speech recognition, it is clear that there is still a long path ahead. First, system performance even on a subset of domains (telephony transcription) – say at best 20% WER (word error rate) – is still a good factor of three or four away from human performance on this task – best reported results being about 5%. Second, except for the preliminary results described in Sect. 30.4.1, no serious attempts have yet been made to develop a single system simultaneously capable of handling transactional and transcription data. Lastly, the robustness of today's speech systems with respect to varying channels and noise is still weak. Is it possible, then, to achieve superhuman speech recognition by plugging away at the essentially incremental approaches that have typically resulted in incremental gains on the order of 10–20% a year that have been described in the rest of the paper, with no major insights from human performance?

Today's speech recognition systems have all evolved to have the same basic structure: spectral features are periodically sampled at the frame level from the speech signal, and the probability distribution of these feature vectors is captured by hidden Markov models (HMMs). The HMMs are combined with a language model of the short-term dependencies of word sequences, and the word hypotheses are produced by various efficient dynamic-programming-based search mechanisms [30.33]. Many alternatives and enhancements have been tried over the years: articulatory models, neural networks, segment models, trigger language models, structured language models, but all have either failed or not shown significant advantages over this basic structure. In addition, regular significant improvements have always resulted over the years via refinements to the basic structure, placing a successively increasing barrier to entry on alternative formulations.

There is no question that advances in speech recognition will still continue via incremental techniques – one would be foolish in the face of essentially 30 years of incremental improvements to deny that progress has not been continually made. However, this does not rule out exploration of alternative methodologies in parallel or on top of existing high-performance recognition systems. In the remainder of this chapter, some speculation is presented both about some promising incremental approaches with good payoffs, as well as some more-dramatic changes in speech recognition methodologies.

30.5.1 Proposed Human Listening Experiments

The listening experiments described above only scratched the surface in terms of trying to tease out what the most useful research directions in speech recognition might be. An extension of the above described experiments is the following.

1. Word sequences are generated automatically from a trigram LM, and then read aloud.
2. The sentences are decoded with an ASR (automatic speech recognition) system, and the WER is computed.
3. Noise is added to the to the recordings until human listeners have the same WER, i. e., the acoustics are calibrated to equalize human and machine performance.
4. Meaningful sentences are recorded, read, and decoded by the ASR system.
5. The same level of noise is added to the same meaningful sentences, which are then transcribed by human listeners.

When meaning is added to the sentences, there will be a differential impact on human and machine performance, presumably with humans benefiting more. This difference will provide a meaningful measure of the amount that can be gained by moving beyond trigram LMs and using broad syntactic, semantic, and pragmatic knowledge sources.

Yet another problem is that there is a mismatch between the metrics that are used to measure ASR performance, and the intended use of these systems. Performance is measured with word error rate, which is dominated by errors in function words. For example, the

five most common errors in a typical large-vocabulary recognition system are:

1. and/in
2. it/that
3. was/is
4. that/the
5. the/a

While these words are very common, it is not clear that recognizing them is always critical to the performance of a real application. For example, in a typical call-center application, a user might call to make a travel reservation and say something like *I want to fly to Boston on Tuesday.* What the system needs to produce is a representation something like (what = reservation; to = Boston; date = Tuesday; from = ????) and realize that an appropriate response is something like *Where are you flying from?* It does not matter, however, if there are minor recognition errors like *I want to fly to Boston at Tuesday,* or even more serious ones like *I went to fly to Boston on Tuesday,* as long as the system is able to recover the underlying meaning with confidence. In order to understand the relationship between word error and concept error, we propose to conduct a final set of experiments to flesh out this relationship.

One such experiment involves revisiting the confusion network experiments described earlier, but in the context of question-answering. In this experiment, subjects are shown confusion networks and then asked to answer questions. If they are able to do this on the basis of the sausages, then we are warranted in taking confusion networks – without further acoustic processing – as the basis for high-level linguistic processing, regardless of WER.

A second experiment revisits the notion of *calibrated acoustics*, but again in the context of question-answering. Here, increasing noise is added to utterances, and humans are asked to both transcribe the utterances and answer questions. Presumably, people will be able to answer questions even after the WER of their transcripts has degraded considerably, indicating that WER is not a good measure of the information transmitted, and suggesting the use of concept error or slot-filling error as a better metric.

In summary, by conducting a series of human listening and comprehension experiments, one can gain a better understanding of how to improve our current systems, what the fundamental limitations of different knowledge sources are, and how to measure system performance.

In addition to human listening experiments, it might be possible to establish other fundamental bounds, e.g., of orders of magnitude extensions of data sets for current basic structure language models. For instance, it is possible to use a Google query interface [30.34] to check the frequency of word sequences appearing in the lattice or *sausage*, i. e., to see if and how often corresponding 5-grams, 4-grams, 3-grams and 2-grams appear in the Google index. Such experiments would help to answer questions about the value of having four orders of magnitude more data for language modeling.

30.5.2 Promising Incremental Approaches

Modeling Spontaneous Speech

The main problem with spontaneous speech is that it contains substantial segmental and suprasegmental variations not present in read speech (which, of course, is much easier to collect in bulk), such as pitch and duration variations, hesitations, false starts, ungrammatical constructs, emotionally expressive speech (laughter, etc.). The high levels of human performance that can be obtained from just auditory presentation of speech out of context [30.4] suggest that the only clues that are needed to decipher speech lie in the local speech signal. In extremely noisy environments, other cues, such as visual ones, may be required. Consequently, if the WER performance is bad, it must be the case that

1. the stochastic models used to model the observation are not accurate;
2. current feature-extraction techniques (mel cepstra) do not extract all the necessary information;
3. the language model is inadequate; or most likely,
4. all of the above.

Most of the speech recognition systems today use frame-level observations. Though some work has been done on suprasegmental feature extraction [30.35], it has enjoyed only a limited amount of success. One possible reason is because neither frame-level observations nor suprasegmental observations are by themselves sufficient to do an adequate job of modeling (spontaneous) speech. Segmental models [30.36, 37] make fewer assumptions than HMMs and provide a better modeling framework by modeling sequences of observations rather than individual observations; however, the goal of multiscale modeling requires a framework that is more powerful than segmental models. The end objective of any model is to compute the joint probability of all observations. This joint probability can be expressed as a product of several conditional probabilities

– however our goal is to make this factorization different for different observations. Graphical models provide a mechanism whereby different factorizations of a joint distribution can be specified by means of a directed graph [30.38].

To model the statistical dependencies in the extracted features a graphical model that specifies the factorization of a joint distribution by means of a graph can be used. This framework can be used to model arbitrary dependencies, including temporal dependencies in different observation streams. Appropriate dependencies can be selected via correlation or mutual information techniques [30.39, 40]. In addition to these statistically identified dependencies, linguistic knowledge related to the speech production process can also easily be incorporated into the graphical model – for instance, the formant frequency estimates at a given time frame could provide valuable information about the identity of the phone at the current and adjacent time frames. Also, constraints on the formant trajectories between adjacent time frames (for continuity reasons) can easily be incorporated into graphical models.

Such a scheme was recently tried [30.41] and demonstrated improvements on the switchboard corpus. A particularly attractive aspect of such a model is that it can easily be augmented with additional suprasegmental features, such as say pitch, which might prove valuable for modeling syllables and words in tonal languages such as Mandarin, or for better spotting of disfluencies.

Multi-Environment Systems

Another objective in achieving human performance is to develop a system that will work on several different types of speech: full bandwidth, telephone bandwidth, or recorded with close-talking or far-field microphones. One possibility is to bandlimit all sources of speech data and train a system using this data, however, this would provide inferior performance for cases where full-bandwidth or low-noise speech is available. Another possibility is just to run multiple models in parallel with a switch that detects bandwidth changes, as is done in many of today's systems that process broadcast news, but that approach completely fragments data into small pieces, generating a complex multicomponent system that is very hard to manage. Consequently, the goal is to either: (i) develop a single modeling framework that can make use of all the information in full-bandwidth speech when it is available, and can also deal with bandlimited speech, or (ii) investigate features that are insensitive to bandwidth-related distortions.

A possible modeling framework for (i) is to partition the observation vector into two components, one of which is always available, $o_{1,t}$, while the other may be hidden, $o_{2,t}$. The formulation is based on the fact that, if the global joint statistics of o_1 and o_2 are known, and for each class the probability density of o_1 and o_2 are known, then it is possible to predict the missing observation from the joint statistics.

The basic speech recognition problem is to find the joint probability of the observation sequence $o_{1,1} \cdots o_{1,T}$, and, when it is available, $o_{2,1} \cdots o_{2,T}$. As the observation probability is conditioned on a state sequence s_1^T that is hidden, this may be written as

$$\begin{aligned} p\left(o_{1,1}^T o_{2,1}^T\right) &= \sum_{s_1^T} p\left(o_{1,1}^T o_{2,1}^T, s_1^T\right) \\ &= \sum_{s_1^T} p\left(o_{2,1}^T / o_{1,1}^T, s_1^T\right) p\left(o_{1,1}^T / s_1^T\right) p\left(s_1^T\right) \end{aligned} \tag{30.2}$$

If $o_{2,1}^T$ is observable, we can approximate the term $p(o_{2,1}^T / o_{1,1}^T, s_1^T)$ by $p(o_{2,1}^T / s_1^T)$, and if $o_{2,1}^T$ is not observable, we can approximate it by $\hat{o}_{2,1}^T$, where $\hat{o}_{2,1}^T = p(o_{2,1}^T / o_{1,1}^T)$, and the joint statistics of $o_{1,1}^T, o_{2,1}^T$ are used to compute the latter term. (For a Gaussian joint distribution, $\hat{o}_2$ turns out to be a linear operation on o_1.) Hence, the global joint statistics of o_1, o_2 are used to predict the value of o_2, making it possible to use information related to the probability distribution function (PDF) of o_2 for the class s, even when o_2 is not observed.

A generalization of this approach was successfully used in [30.42] to improve robustness for speech in very high-noise environments. In such noise, large sections of the time–frequency display for a speech signal are often masked by the noise, not just the higher frequencies as in telephony speech. It was shown that variations on the above technique could improve the effective signal-to-noise (S/N) ratio by more than 10 dB in certain cases. However, the technique has not yet been applied to bandlimited speech.

Possible frameworks for (ii) include the following. [30.40] investigated the utility of a *spectral-peak* feature that is closely related to the formant frequencies, and initial experimental results indicate that the spectral peak features definitely carry information that can help speech recognition even in clean conditions. These initial experiments were based on simple-minded *feature fusion* of the spectral peak features with the cepstra, and going beyond this simple approach to the more-sophisticated multiscale graphical models described above, which en-

able trajectories to be modeled in the formant space, will lead to powerful and robust models for speech recognition.

Another approach to (ii) is to adapt the models or features to the new environmental conditions. In addition to popular techniques such as MLLR and FMLLR, and more recently, fMPE, another approach is to use a nonparametric mapping approach for the cumulative probability distribution function [30.43]. Assume that it is possible to transform the adaptation features $x' = g(x)$ in such a way that the cumulative distribution function (CDF) of the transformed adaptation data is made identical to the CDF of the training data. This criterion is used to guide the design of the transformation. Falling as it does into the category of *data transformation* techniques, this method is independent of the modeling framework that is used; consequently, it can be easily incorporated into the segmental graphical model framework suggested earlier. In [30.44], this method was applied to the speech samples for the application of speaker identification; here this method is extended to deal with multidimensional observations that are used in speech recognition.

This approach is motivated by the following reasoning. It is well known that the assumed parameterization of the true PDF of the data is often inaccurate. Consequently, if we could deal with the empirically observed CDF of the data, there would be no need to make any modeling assumptions. Secondly, the CDF is a well-behaved function (monotonic and lying between 0 and 1), consequently it is relatively easy to define a mapping of the feature dimension that equates two CDFs. Thirdly, this method is computationally much simpler than most other adaptation schemes. Most adaptation schemes require the adaptation data from the test environment to be transcribed, with an associated overhead. The CDF matching scheme on the other hand does not require any prior transcription, and the process of computing the nonlinear transformation is also relatively inexpensive.

For the case of multidimensional features, if the dimensions are independent, then the CDF-matching technique can be applied to each dimension independently. However, as this is not generally the case, it is necessary first to transform the feature into a space where the dimensions are uncorrelated. This needs to be done simultaneously for both the training and adaptation data. The solution to this problem is a transformation consisting of the generalized eigenvectors of the covariance matrices of the training and test data.

This technique has been been used to compensate for the mismatch between landline and speakerphone telephone data. In Fig. 30.4, the CDF for the first cepstral dimensions for landline and speakerphone data is shown, with the corresponding mapping. In preliminary experiments, the CDF mapping technique improved performance on speakerphone data by 30% relative [30.43].

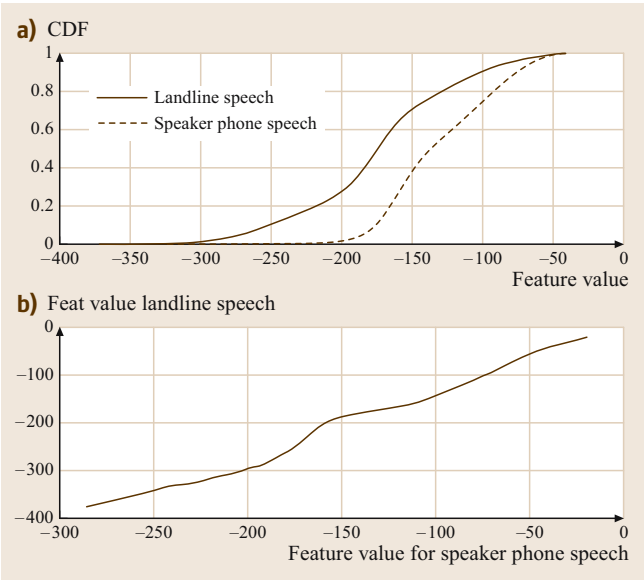


Fig. 30.4a,b CDF matching for landline and speakerphone features. **(a)** CDF functions for landline and speakerphone data, **(b)** the input–output mapping

Improved Language Modeling

In the last few years, a number of research results by IBM [30.45, 46] and other research groups [30.47–49] have appeared that strongly suggest that linguistic information, such as meaning and language structure, can be used to improve speech recognition performance significantly. The scale of these research activities and experiments is limited and recognition experiments have tended to be performed only on narrow domains. However, the improvements are encouraging. For example, the semantic structured language models [30.45, 46] shown in Fig. 30.5 have reduced WER by 20%, and increased confidence by 30–60% for multiple ASR task domains, such as air travel, finance, medical and military domains, compared to when no semantic structure information was used. In such a language model, the N-best hypotheses are first produced by the recognizer. For each hypothesis, the recognized words are then mapped onto a set of *classes*. Then, a semantic parse of the classed words in each hypothesis is produced, and a maximum-entropy model is constructed in which the word w_j is

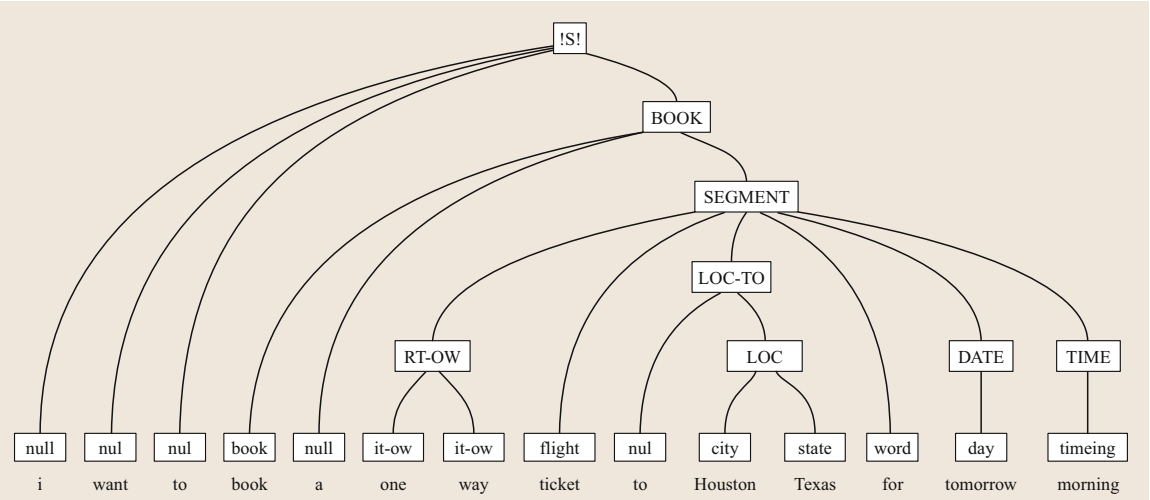


Fig. 30.5 A semantic structured language model. The words are shown at the *bottom* of the parse tree. The next level up represents the output of the word classer, and the higher levels represent the parse tree constituents. At the bottom is shown a text-based bracketed representation useful for determining language model dependencies (see text)

predicted as

$$p\left(w_j|W_1^{j-1}\right) = p(w_j|w_{j-1}, w_{j-2}, p_j, g_j, c_j)\,, \tag{30.3}$$

where p_j is the parent constituent of the current word, c_j is the last complete constituent, and g_j is the constituent identity of the highest-level concept below the sentence level in the semantic parse tree. For example, in Fig. 30.5 the parent constituent of *tomorrow* is *DATE*, the last complete constituent for *morning* is *DATE*, and the highest level concept for *morning* is *BOOK* (note that the lower boxes are the class assignments and not the parse tree constituents). The resultant language model is then used to rescore the N-best hypotheses.

In these experiments, only a small number of linguistic features, among the almost unlimited number of them, have been investigated for speech recognition. Typically, they are restricted to extracting syntactic or semantic attributes, and fitting them into the basic structure of speech recognition, i. e., adding semantic or structural attributes into a trigram or bigram language model and computing the new perplexity or measuring the resulting WER. These research activities are mainly in the area of implementation or engineering, rather than in developing new computational models that can naturally marry automatic speech recognition and computational linguistics. Only the latter can be expected to result in dramatic ASR performance improvements.

Such efforts should come naturally, as the types of information, i. e., acoustic (mainly statistical) and linguistic (mainly structural), are complementary. Even though creating complete new models for speech recognition is difficult, there are examples of leveraging the combined power of statistical and structural information – namely, in the field of natural language processing. Taking insights from such areas of computational linguistics such as word sense disambiguation, text meaning representation, and syntactic parsing, and combining them with the power of statistical techniques, new models have been created and successfully applied in text analytics, search and classification. Often these models use new knowledge resources, such as the Penn TreeBank, the Brown Corpus, WordNet, and the extended WordNet.

Based on these observations, two parallel lines of research are suggested in order to incorporate semantic and syntactic knowledge into ASR:

1. Investigate the use of a larger number of linguistic features, such as lexical features, semantic and syntactic relation and structural features, morphological features, part–whole relationship features, and experiment with more task domains of ASR. Gradually extend to broad domains of ASR. These will test the broad validity of the claims about the importance of semantic and structural information as well as help to find the set of useful features.
2. Investigate new computational models for speech recognition in which natural language structures and linguistic features are naturally integrated in one unified framework. The direct model proposed by IBM [30.50] is one possible approach. In this framework, a model is constructed in which the probability of a phonetic state, word, or sentence is computed as a direct function of the underlying knowledge sources including both linguistic and acoustic information. Figure 30.6 depicts such a model in which the probability of the next state of the model is a function of the word, the sentence, and the acoustic observation. This is contrasted with current practice, which uses a generative model to compute the probability of a set of features from an underlying word or sentence in conjunction with an independent language model. The new models could be used as one of the specialists in the doctor/specialist framework described below.

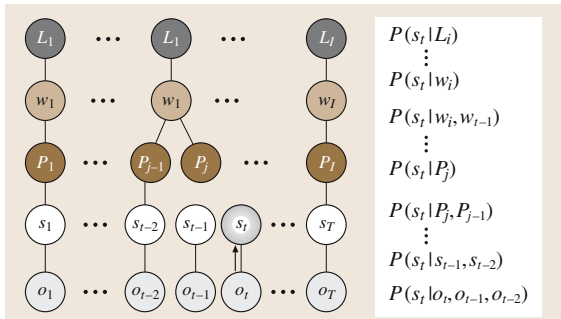


Fig. 30.6 A direct model for automatic speech recognition. Here, the probability of the state is directly conditioned on the associated phoneme, the word, the language, and the observations

30.5.3 Promising Disruptive Approaches

The Doctor/Specialists Paradigm

State-of-the-art ASR systems adopt a monolithic approach to acoustic and linguistic knowledge sources. For example, the same acoustic features and model are used to distinguish all acoustic confusions [30.33]. It is quite remarkable that this approach to ASR, chosen for its simplicity and lack of a better alternative, works so well in practice. However, it is also clear that for distinguishing particular acoustically confusable units, e.g., *a* versus *the*, a specialist classifier trained only to distinguish between the confusable units will perform significantly better. One solution is the design and development of a *doctor/specialists* framework for ASR. The basic idea is as follows. The doctor, who is responsible for the overall recognition process, will partition the speech into regions of certainty and regions of confusion, which can be done with the current state of the art. In regions of confusion, inputs from specialists (acoustic, linguistic, and visual) who are experts at distinguishing between particular sets of words or phonemes will be invoked and integrated in a systematic manner.

A key ingredient in ASR improvements over the past decade has been the inclusion of increasing amounts of contextual information – both acoustic and linguistic, and the framework above can be viewed as a significant step towards exploiting larger contextual information. The doctor uses all the contextual information available at a given point to decide on what the next local problem to be resolved is (e.g., is it *f* as in *fine* or *sh* as in *shine*) and then consults the best specialist for this problem. This approach to ASR may be significantly better than the current approach for the following reasons. First, specialists can use more-complex and discriminative models than those in use today [30.51, 52]. Specifically, for the specialists one can replace the generative (HMM-based) models in use today with the discriminative models that have been hard to use in ASR because of the large number of acoustic classes being modeled simultaneously. Secondly, specialists need only be trained in regions difficult for the doctor. Thirdly, specialists can focus on novel acoustic, linguistic and visual (when available) features specific to the confusions they have to resolve. Finally, because of the smaller size of the problems they tackle, specialists can use computationally and memory intensive template-matching approaches for resolving confusions. This methodology may also entail using orders of magnitude more acoustic training data than used by current systems (e.g., 100 000 h of speech). Some initial promising results on such a technique has been obtained in [30.53], where

the term *acoustic codebreaking* is used to describe the doctor/specialists procedure.

In principle this doctor/specialists approach outlined requires a completely new search strategy (doctor) and associated modeling strategies (specialists). In practice, we believe this strategy is well suited for implementation in the context of the current state of the art. Current ASR systems can generate confidence-weighted word lattices or sequences of word confusion sets (or sausages, see Fig. 30.1) [30.29]. The oracle word error rate on lattices and/or sausages can be significantly lower than the one-best word error rate, e.g., 10% versus 30%. This implies that resolving confusions in lattices or sausages will result in a dramatic gain in accuracy. To exploit this one would require

- research on the construction of acoustic, linguistic, and visual specialists to distinguish between phonemes and/or words that typically co-occur in sausages
- research on when to invoke and how to integrate input from the specialists

This manifestation of this paradigm naturally decouples the recognition process into two tasks, each of which can be the thrust of independent research programs:

- the generation of minimal size lattices and/or sausages with zero oracle word error rate, and
- the choice of the one-best word hypothesis in lattices and/or sausages with the help of specialists adapted to the particular confusions present in a given lattice or sausage.

Large-Scale Information Fusion

The current state of the art in speech analysis operates in a highly stylized way: the waveform is processed in 25 millisecond chunks – one every 10 milliseconds – regardless of the acoustic conditions, and linguistic knowledge is captured at the level of word N-grams only. It is known, however, that in animal auditory processing there are hundreds of cells and neural regions that are specially adapted to respond to specific acoustic and linguistic stimuli [30.54], and that somehow these features are combined on an extremely large scale to produce speech perception. Drawing loosely on this analogy, another disruptive approach to speech recognition is to develop methods for extracting and combining extremely large numbers of complimentary acoustic and linguistic features. Figure 30.7 illustrates some of the acoustic aspects of this paradigm.

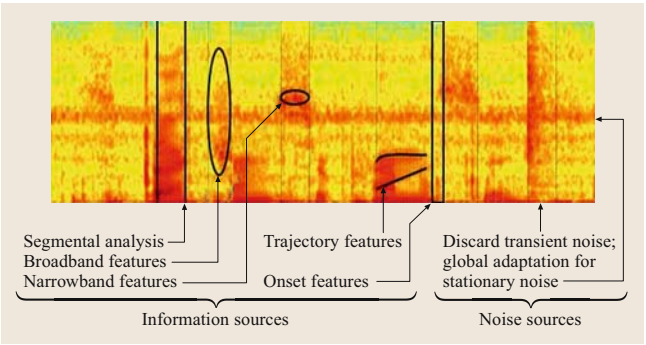


Fig. 30.7 Multiscale and variable-rate features

Here, the spectrum corresponding to a word has been segmented into variable-length components, both vertically and horizontally. The first oval represents a broadband feature measuring the energy across a wide range of frequencies. The second oval, in contrast, is a narrowband feature, while the dark rectangle at the right measures the rate of change of energy across all frequencies, resulting in an onset detector. The individual curved lines track the formants of one syllable, and their temporal evolution results in two trajectory features. The long horizontal band of orange in the middle of the figure is hypothesized to be noise, and adaptation is used to remove its effects. Once a large number of features has been defined and evaluated, it still remains to combine them in a coherent probabilistic model, and here the maximum-entropy approach is ideally suited. The values of arbitrary numbers of both acoustic and linguistic features can be combined in a theoretically sound fashion to produce posterior estimates of word probabilities.

Although conceived as using far more features, and combining them in a less-supervised way, this model is closely related to the doctor/specialists model. The integrating probabilistic model, e.g., maximum entropy, acts essentially as a word-level doctor and combines features to determine word identity on a word-by-word basis. The feature values represent specialist outputs, and maximum entropy provides a framework for weighting them.

Large-Scale Decoder Combination

Recent competition-grade ASR systems have begun to combine the outputs of several systems in a voting process [30.55]. In this strategy, a system actually consists of four or five independently constructed ASR system. Each of these is handcrafted to differ from the others in some minor details, for example the use of MFCC as opposed to PLP features, or the set of phonetic units

that is used. Typically these systems make somewhat uncorrelated errors, and therefore a voting strategy tends to converge on the correct answer. Often, the systems represent a failed attempt at building a better single decoder that results simply in something that makes equally many but different errors.

This basic strategy has been proven to create gains, but ignores one of the major developments in machine learning – the idea, termed boosting, that hundreds or thousands of systems can actually be manufactured and combined in a fully automated way so as to create mathematically provable bounds on the error rate of the resulting composite [30.56]. In this framework, a sequence of classifiers is constructed such that at each iteration the latest classifier tends to correct the errors of the previous ones. This is done by maintaining a carefully selected weight on each of the training examples so that it is possible to prove that, as long as a better-than-chance classifier can be built at each iteration, the error rate will drop to zero in an exponentially decreasing fashion. On the surface this may appear to generate a system of extreme complexity: a 40 dB complexity increase relative to the system described in Sect. 30.4.2. The key challenge is to develop a set of systems automatically that all have a similar structure whose overall management is similar. An initial start in this direction was described in [30.57], in which a set of similar systems were generated from the same data by randomizing the data used to initialize the state context acoustic decision trees. Significant recognition improvements were demonstrated on the switchboard task by combining up to six similar randomized systems. This can be generalized to other components of the system – say the signal processing and language models – to generate many parallel systems. Of course, boosting is not a panacea and a major issue will be how to generate appropriate sets of complementary systems to be combined.

In order to achieve such goals, the computing power of large clusters of workstations – using new architectures such as Blue Gene [30.58] and Cell [30.59] – can be used to build and combine decoders on a scale that has not been conceived before. Doing this will require full parameterization of the process of building a decoder (within the basic structure) so that system builds that utilize different subsets of the training data will be able to span the necessary space of acoustic and language models. Perhaps in the end this is the most promising approach of all, insofar as it begins to address building systems that even approach the complexity and processing power of the human brain.

References

30.1 J.G. Fiscus, W.M. Fisher, A.F. Martin, M.A. Przybocki, D.S. Pallett: 2000 NIST evaluation of conversational speech recognition over the telephone, Proc. 2000 Speech Transcription Workshop (2000)

30.2 A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, C. Wooters: The ICSI Meeting corpus, Proc. ICASSP, Vol. I (2003) pp. 364–367

30.3 M. Padmanabhan, G. Saon, J. Huang, B. Kingsbury, L. Mangu: Automatic speech recognition performance on a voicemail transcription task, IEEE Trans. Speech Audio Process. **10**(7), 433–442(2002)

30.4 R.P. Lippmann: Speech recognition by machines and humans, Speech Commun. **22**(1), 1–15 (1997)

30.5 I. Pollack, J.M. Pickett: The intelligibility of excerpts from conversation, Lang. Speech **6**, 165–171 (1963)

30.6 E. Chang, R. Lippmann: Improving wordspotting performance with artificially generated data, Proc. ICASSP, Vol. 1 (1996) pp. 526–529

30.7 J.B. Allen: How do humans process and recognize speech?, IEEE Trans. Speech Audio Process. **2**(4), 567–577 (1994)

30.8 C.E. Shannon: Prediction and entropy of printed English, Bell Syst. Tech. J. **30**, 50–64 (1950)

30.9 NIST Speech Group: The Rich Transcription Spring 2003 (RT-03S) Evaluation Plan, Version 4 (2003) http://www.nist.gov/speech/tests/rt/rt2003/spring/docs/rt03-spring-eval-plan-v4.pdf

30.10 W. Byrne, D. Doermann, M. Franz, S. Gustman, J. Hajič, D. Oard, M. Picheny, J. Psutka, B. Ramabhadran, D. Soergel, T. Ward, W.-J. Zhu: Automatic recognition of spontaneous speech for access to multilingual oral history archives, IEEE Trans. Speech Audio Process. **12**(4), 420–435 (2004)

30.11 I. Nastajus: http://en.wikipedia.org/wiki/Naturally Speeking (2007)

30.12 P. Woodland, H.Y. Chan, G. Evermann, M.J.F. Gales, D.Y. Kim, X.A. Liu, D. Mrva, K.C. Sim, L. Wang, K. Yu, J. Makhoul, R. Schwartz, L. Nguyen, S. Matsoukas, B. Xiang, M. Afify, S. Abdou, J.-L. Gauvain, L. Lamel, H. Schwenk, G. Adda, F. Lefevre, D. Vergyri, W. Wang, J. Zheng, A. Venkataraman, R.R. Gadde, A. Stolcke: SuperEARS: Multi-Site Broadcast News System, DARPA EARS 2004 Workshop (2007), http://www.sainc.com/richtrans2004/uploads/monday/EARS BN Super team.pdf

30.13 A. Aaron, S. Chen, P. Cohen, S. Dharanipragada, E. Eide, M. Franz, J.-M. Leroux, X. Luo, B. Maison, L. Mangu, T. Mathes, M. Novak, P. Olsen, M. Picheny, H. Printz, B. Ramabhadran, A. Sakra-

jda, G. Saon, B. Tydlitat, K. Visweswariah, D. Yuk: Speech recognition for DARPA Communicator, Proc. ICASSP, Vol. 1 (2001) pp. 489–492
30.14 H. Soltau, B. Kingsbury, L. Mangu, D. Povey, G. Saon, G. Zweig: The IBM 2004 conversational telephony system for rich transcription, Proc. ICASSP, Vol. 1 (2005) pp. 205–208
30.15 J. Fiscus: The Rich Transcription Spring 2006 (RT-06S) Evaluation Results (NIST Speech Group, 2007) http://www.nist.gov/speech/tests/rt/rt2006/spring/pdfs/rt06s-STT-results-v7.pdf
30.16 G. Saon, M. Padmanbhan, R. Gopinath, S. Chen: Maximum likelihood discriminant feature spaces, Proc. ICASSP, Vol. II (2000) pp. 1129–1132
30.17 R.A. Gopinath: Maximum likelihood modeling with Gaussian distributions for classification, Proc. ICASSP, Vol. 2 (1998) pp. 661–664
30.18 M.J.F. Gales: *Semi-tied full-covariance matrices for hidden Markov models*, Vol. CUED/F-INFENG/TR287 (Cambridge Univ. Engineering Department, Cambridge 1997)
30.19 J. Huang, B. Kingsbury, L. Mangu, G. Saon, R. Sarikaya, G. Zweig: Improvements to the IBM hub 5e system, Proc. NIST RT-02 Workshop (2002)
30.20 G. Saon, G. Zweig, B. Kingsbury, L. Mangu, U. Chaudhari: An architecture for rapid decoding of large vocabulary conversational speech, Proc. Eurospeech, Vol. 3 (2003) pp. 1977–1981
30.21 S. Axelrod, V. Goel, B. Kingsbury, K. Visweswariah, R. Gopinath: Large vocabulary conversational speech recognition with a subspace constraint on inverse covariance matrices, Proc. Eurospeech, Vol. 3 (2003) pp. 1613–1616
30.22 S. Axelrod, R.A. Gopinath, P. Olsen: Modeling with a subspace constraint on inverse covariance matrices, Proc. Int. Conf. Spoken Lang. Process., Vol. 2 (2002) pp. 2177–2180
30.23 S. Wegmann, D. MacAllaster, J. Orloff, B. Peskin: Speaker normalization on conversational telephone speech, Proc. ICASSP, Vol. 1 (1996) pp. 339–342
30.24 M.J.F. Gales: *Maximum likelihood linear transformations for HMM-based speech recognition*, Vol. CUED/F-INFENG/TR291 (Cambridge Univ. Engineering Department, Cambridge 1997)
30.25 C.J. Leggetter, P.C. Woodland: Speaker adaptation of continuous density HMMs using multivariate linear regression, Proc. Int. Conf. Spoken Lang. Process., Vol. I (1994) pp. 451–454
30.26 S.F. Chen, J. Goodman: An empirical study of smoothing techniques for language modeling, Computer, Speech Lang. **13**(4), 359–393 (1999)
30.27 L.R. Bahl, P.V. deSouza, P.S. Gopalakrishnan, D. Nahamoo, M. Picheny: Robust methods for using context-dependent features and models in a continuous speech recognizer, Proc. ICASSP, Vol. I (1994) pp. 533–536
30.28 M. Padmanabhan, G. Ramaswamy, B. Ramabhadran, P.S. Gopalakrishnan, C. Dunn: Issues involved in voicemail data collection, Proc. DARPA Broadcast News Transcription and Understanding Workshop (1998)
30.29 L. Mangu, E. Brill, A. Stolcke: Finding consensus in speech recognition: Word error minimization and other applications of confusion networks, Computer, Speech Lang. **14**(4), 373–400 (2000)
30.30 E. Shriberg, A. Stolcke, D. Baron: Observations on overlap: Findings and implications for automatic processing of multi-party conversation, Proc. Eurospeech, Vol. 2 (2001) pp. 1359–1362
30.31 D. Povey, P. Woodland: Minimum phone error and I-smoothing for improved discriminative training, Proc. ICASSP, Vol. 1 (2002) pp. 105–108
30.32 D. Povey, B. Kingsbury, L. Mangu, G. Saon, H. Soltau, G. Zweig: FMPE: Discriminatively trained features for speech recognition, Proc. ICASSP, Vol. 1 (2005) pp. 961–964
30.33 M. Padmanabhan, M. Picheny: Large-vocabulary speech recognition algorithms, IEEE Comput. **35**(4), 42–50 (2002)
30.34 Google Desktop Developer Group: http://www.google.com/apis/ (2007)
30.35 B.E.D. Kingsbury, N. Morgan, S. Greenberg: Robust speech recognition using the modulation spectrogram, Speech Commun. **25**(1-3), 117–132 (1998)
30.36 M. Ostendorf, V.V. Digilakis, O.A. Kimball: From HMMs to segment models: A unified view of stochastic modeling for speech recognition, Proc. IEEE Trans. Speech Audio Process. **4**(5), 360–378 (1996)
30.37 J. Bridle, L. Deng, J. Picone, H. Richards, J. Ma, T. Kamm, M. Schuster, S. Pike, R. Regan: An investigation of segmental hidden dynamic models of speech coarticulation for automatic speech recognition, Final Workshop Report, Center for Language and Speech Processing (The Johns Hopkins University, Baltimore 1998)
30.38 G. Zweig, M. Padmanabhan: Dependency modeling with Bayesian networks in a voicemail transcription system, Proc. Eurospeech, Vol. 3 (1999) pp. 1335–1338
30.39 J. Bilmes: Buried Markov models, Proc. ICASSP, Vol. 2 (1999) pp. 713–716
30.40 M. Padmanabhan: Use of spectral peak information in speech recognition, Proc. NIST Speech Transcription Workshop (2000)
30.41 Ö. Çetin, M. Ostendorf: Multi-rate and variable-rate modeling of speech at phone and syllable time scales, Proc. Int. Conf. Acoust. Speech Signal Process., Vol. 1 (2005) pp. 665–668
30.42 M.P. Cooke, P.D. Green, L.B. Josifovski, A. Vizinho: Robust automatic speech recognition with missing

and uncertain acoustic data, Speech Commun. **34**, 267–285 (2001)
30.43 S. Dharanipragada, M. Padmanabhan: A nonlinear unsupervised adaptation technique for speech recognition, Proc. Int. Conf. Spoken Lang. Process., Vol. IV (2000) pp. 556–559
30.44 R. Balchandran, R. Mammone: Non-parametric estimation and correction of non-linear distortion in speech systems, Proc. ICASSP, Vol. II (1998) pp. 749–752
30.45 H. Erdogan, R. Sarikaya, Y. Gao, M. Picheny: Semantic structured language models, Proc. Int. Conf. Speech Lang. Process., Vol. II (2002) pp. 933–936
30.46 R. Sarikaya, Y. Gao, M. Picheny: Word level confidence measurement using semantic features, Proc. ICASSP, Vol. I (2003) pp. 604–607
30.47 J. Bellegarda: Exploiting latent semantic information in statistical language modeling, Proc. IEEE **88**(8), 1279–1296 (2000)
30.48 F. Jelinek, C. Chelba: Putting language into language modeling, Proc. Eurospeech, Vol. 1 (1999) pp. KN–1–KN–4
30.49 I. Gurevych, R. Malaka, R. Porzel, H.P. Zorn: Semantic coherence scoring using an ontology, Proc. HLT-NAACL (2003) pp. 88–95
30.50 A. Likhododev, Y. Gao: Direct models for phoneme recognition, Proc. ICASSP, Vol. 1 (2002) pp. 89–92
30.51 V. Vapnik: The support vector method, Proc. Int. Conf. Artif. Neural Networds (1997) pp. 263–271
30.52 S. Della Pietra, V. Della Pietra, J. Lafferty: Inducing features of random fields, IEEE Trans. Pattern Anal. Mach. Intell. **19**(4), 380–393 (1997)
30.53 V. Venkataramani, W. Byrne: Lattice segmentation and support vector machines for large vocabulary continuous speech recognition, Proc. ICASSP, Vol. 1 (2005) pp. 817–820
30.54 L. Miller, M. Escabi, H. Read, C. Schreiner: Spatiotemporal receptive fields in the lemniscal auditory thalamus and cortex, J. Neurophysiol. **87**, 516–527 (2001)
30.55 J.G. Fiscus: A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER), Proc. IEEE Workshop Autom. Speech Recognition Understanding, Santa Barbara (1997) pp. 347–355
30.56 Y. Freund, R.E. Schapire: Experiments with a new boosting algorithm, Proc. ICML (1996) pp. 148–156
30.57 O. Siohan, B. Ramabhadran, B. Kingsbury: Constructing ensembles of ASR systems using randomized decision trees, Proc. ICASSP, Vol. 1 (2005) pp. 197–200
30.58 IBM Research Communication Dept.: http://www.research.ibm.com/bluegene (2007)
30.59 IBM Research Communication Dept.: http://www.research.ibm.com/cell (2007)

32. Transcription and Distillation of Spontaneous Speech

S. Furui, T. Kawahara

Automatic transcription of spontaneous human-to-human speech is expected to expand the applications of speech technology, enabling efficient access to audio archives such as broadcast programs, lectures, and meetings. Compared with utterances in human–machine interfaces, which have been the focus of most conventional speech recognition research, spontaneous speech has greater variation in both its acoustic and linguistic characteristics. Therefore, it is necessary to explore more elaborate and flexible modeling techniques for spontaneous speech recognition. Moreover, spontaneous speech processing requires the development of a different paradigm, in that faithful transcription is not necessarily useful because of the existence of disfluencies and the lack of sentence and paragraph markers. Studies on automatic detection of sentence/discourse boundaries and disfluencies are also needed. Speech summarization is an approach that generates effective output from a transcript. This chapter gives an overview of major research activities and a number of recent findings on these topics.

32.1 Background

Automatic speech recognition (ASR) technology has achieved accuracy levels exceeding 90% for read speech and similar styles of speech, including news broadcasts given by anchors. However, the performance of ASR drops off dramatically for spontaneous speech such as human-to-human conversations. Spontaneous speech is significantly different from read speech, both acoustically and linguistically. Spontaneous speech is not clearly articulated with orthodox pronunciation, and contains numerous disfluencies and colloquial expressions. Although human beings face similar problems in hearing foreign languages, current ASR systems se-

riously lack the coverage and flexibility required to cope with spontaneous speech. Thus, in addition to noise robustness, spontaneous speech presents one of the greatest challenges in ASR research.

It is quite interesting to note that, although speech is almost always spontaneous in daily life, spontaneous speech recognition emerged as a visible research theme only about 10 years ago. Most conventional ASR systems have been developed as human–machine interfaces, primarily for the purposes of information input and access. Although these will no doubt remain the primary applications of ASR, there is also a growing interest in viewing speech not simply as a means to access information, but as a source of information in itself. Since speech is the most natural and effective method of communication between human beings, a variety of speech documents, including lectures, meetings, and broadcast programs, are produced everyday. With the pervasion of digital media technologies, these audio materials are increasingly archived both personally and publicly. Even more personal recordings, such as voice messages and conversations, are also being digitally recorded. However, it is not easy to quickly review, retrieve, selectively disseminate, and reuse these speech archives, if they are simply stored as audio signals. Automatic speech transcription is essential for creating knowledge resources from huge speech archives, and thus improving ASR so that it can better cope with spontaneous speech is critical to broaden its potential applications.

For improved ASR performance, it is indispensable to build acoustic and language models for spontaneous speech. Since the characteristics of spontaneous speech are different from those of read-style speech, it may be inadequate to simply apply conventional modeling schemes such as hidden Markov models (HMMs) and n-gram language models. The variation of spontaneous speech is much larger according to the speaker and the speaking style. Thus, we need to investigate what kinds of factors actually affect the characteristics of spontaneous speech and how to parameterize and computationally model these factors.

Despite the wide variety of spontaneous speech, it is difficult and costly to prepare a large speech corpus, because it involves manual transcription of utterances with many disfluencies, compared to the reading of prepared materials. This motivated us to build the corpus of spontaneous Japanese (CSJ), which can be used as a shared resource for various fields of spontaneous speech research. Several works presented herein are based on this corpus.

Spontaneous speech recognition also requires a paradigm shift from conventional ASR, which decodes utterances into a word sequence, because simple transcription of spontaneous speech includes many disfluencies, and so is neither readable nor appropriate for documentation. In addition, sentence and paragraph endings are not explicitly given. Therefore, a number of research projects, including those sponsored by the Defense Advanced Research Projects Agency (DARPA) and the National Institute of Standards and Technology (NIST), are oriented to *rich transcription*, which also involves annotation of end-of-sentence markers and disfluency phenomena. Furthermore, we explore semantic-level processing to extract messages from spontaneous speech such as key sentence indexing and speech summarization. These issues are also addressed herein.

32.2 Overview of Research Activities on Spontaneous Speech

32.2.1 Classification of Spontaneous Speech

Speech recognition tasks can be classified into four categories, as shown in Table 32.1, according to two criteria: whether the targeting utterances are human-to-human or human-to-machine, and whether they constitute a dialogue or a monologue. The table lists typical tasks for each category.

The majority of commercial application systems are classified as category III, recognizing utterances in human-to-machine dialogues. Unlike other categories, the category III systems are usually designed to complete a specific task in a limited domain, for example, quoting stock prices or making ticket reservations. Therefore, the ASR system typically consists of a medium-size vocabulary and a task-specific grammar.

The most typical task belonging to category IV, which targets recognition of monologues performed when people are talking to a computer, is dictation. This task was an initial attempt at achieving general-purpose large-vocabulary continuous speech recognition (LVCSR), and provides a technological basis for other ASR applications, including those of categories I and II. The dictation system, however, assumes that users

Table 32.1 Categorization of speech recognition tasks

	Dialogue	Monologue
Human to human	(Category I) Telephone conversation Panel discussion Meeting	(Category II) Broadcast news Lecture Voice mail
Human to machine	(Category III) Database query Information retrieval	(Category IV) Dictation WSJ task

clearly utter grammatically correct sentences with orthodox pronunciation. This assumption generally holds in human–machine interfaces, especially when users are aware that they are talking to a computer and ASR results are fed back to them in real time.

Typical tasks belonging to categories I and II, recognizing human-to-human interactions, are plotted in Fig. 32.1 with respect to the number of participants and spontaneity. Major projects conducted thus far in these categories are briefly described in the following subsection.

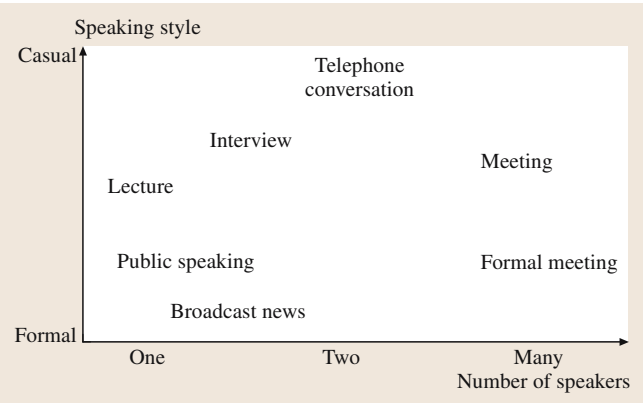


Fig. 32.1 Spontaneous speech recognition tasks

32.2.2 Major Projects and Corpora of Spontaneous Speech

Broadcast News

Broadcast news was initially and is still now intensively targeted by many projects, including DARPA Hub-4 [32.1] and the GALE (global autonomous language exploitation) programs. Most of the utterances in broadcast news programs are made by professional anchors, and language models can be well developed based on a large amount of closed captions. Thus, this particular ASR task is relatively easier. Because broadcast news is a rich information source, it is also used for research on content-based processing, such as topic detection and summarization [32.2, 3].

Oral Presentations and Lectures

Public speaking and oral presentations are also appropriate targets for automatic transcription and archiving. Although the speakers are not necessarily professionals, they generally try their best to be understandable by their audience. Since the topic and vocabulary are often technical, it is necessary to adapt the language model as well as the acoustic model for every speaker. Although there were some previous efforts in this direction such as the TED (Translanguage English Database) corpus [32.4], a recorded collection of EUROSPEECH conference, the most comprehensive one is the corpus of spontaneous Japanese (CSJ) [32.5], which is a 660-hour speech collection of academic presentations and simulated public speeches, recorded using a close-talking microphone. The word error rate (WER) is around 20% with this corpus. The CSJ is thoroughly described in Sect. 32.2.4.

Lectures at universities are also being digitally archived. Their automatic transcription and indexing is also studied at MIT [32.6] and Microsoft under the iCampus project, using hundreds of hours of lectures and seminars at MIT. Similar efforts are being conducted at Kyoto University and the Tokyo Institute of Technology in Japan. The WER for these lectures is 40–50% [32.6]. Unlike the CSJ, the lecturers do not use headset microphones, and the speaking style is less formal in the classroom.

Interviews and Telephone Conversations

Interviews and telephone conversations are basically forms of dialogue, but the interview is a less private style and is similar to the monologue. The MALACH (multilingual access to large spoken archives) project [32.7] was organized to advance the speech technology for access to large multilingual archives assembled by the Survivors of the Shoah Visual History Foundation (VHF). It is a 116 000-hour collection of interviews conducted in 32 languages with nearly 52 000 survivors of the Holocaust. Automatic transcription has been added to portions of the English and Czech speech using ASR with a WER below 40% [32.7]. This task is challenging because speech is heavily accented by elderly speakers who are often emotional.

Dialogue conversations over telephone channels have been most intensively studied in the DARPA-funded projects of HUB-5 [32.8] and EARS (effective affordable reusable speech-to-text) [32.9, 10]. The

Switchboard corpus [32.11], the CALLHOME corpus, and the Fisher corpus were set up for this purpose. Telephone conversations are generally the most spontaneous and casual. As a result, acoustic and pronunciation variations are very prominent. In terms of linguistic aspects, it is observed that a small vocabulary, characteristic of conversational style, accounts for large coverage of utterances, although these words have many pronunciation variants. The WER had been very large (around 40%), but was improved to 15% with the increase of the training data to 2100 hours and the combination of various training and decoding techniques [32.9, 10, 12].

Meetings

Meetings are also emerging as a major target of spontaneous speech recognition. Meeting recognition projects were initiated by NIST [32.13] and several European-funded projects such as AMI (augmented multi-party interaction) and CHIL (computers in the human interaction loop). Since meetings involve multiple participants, it is necessary to determine who spoke when. This task is referred to as speaker diarization. Overlapping of utterances is often observed and severely affects both speaker diarization and speech recognition. Moreover, the effect of the use of distant microphones, including microphone arrays, is also investigated. The WER is 20–30% with close-talking microphones and 30–40% with distant microphones [32.14]. Since the spontaneity of utterances is high, it was shown that the incorporation of telephone conversation corpora was effective when the size of the matched training data was small.

More formal meetings include those in Congress or Parliament. In Europe, the TC-STAR (technology and corpora for speech to speech translation) project [32.15], funded by the European Commission, primarily targets automatic speech transcription and translation of the meetings in European Parliament plenary sessions. The WER is close to 10% [32.16]. In Japan, the National Congress (Diet) is interested in the introduction of ASR technology into the next generation of the meeting transcription system. There, most of the meetings are not in plenary sessions, but in committees where the degree of spontaneity is higher. The WER without using matched training data was around 20% [32.17], and is improved to around 15% with a matched corpus.

32.2.3 Issues in Design of Spontaneous Speech Corpora

The appetite of today's statistical speech processing techniques for training material is well described by the aphorism, *There's no data like more data*. Large collections of speech and text, or corpora, are vital for research and development of ASR. Statistical methodology provides an automatic procedure to directly *learn* regularities in the speech data, but this means that the performance of the derived models and systems are significantly affected by the characteristics and the quality of the corpora. Therefore, there are several issues to be considered in the design of spontaneous speech corpora.

The first problem is how to collect speech by controlling variety and spontaneity. It is desirable to collect natural spontaneous speech without any constraints, but there are always privacy or corporate property issues encountered in collecting real data in public or in companies. Thus, we often ask collected subjects to speak naturally in a given situation or on a given topic. In this case, selection of the topic and the intimacy of the participants deeply affect the spontaneity. The variety of speakers and vocabulary must also be taken into account to make a general corpus.

The second issue is the quality of transcription and annotation. Manual transcription of spontaneous speech involves huge labor costs. Moreover, spontaneous speech includes many disfluency phenomena, such as self-repairs and partial words, and so specific guidelines for annotation are needed. For example, how do we transcribe when speakers make ambiguous pronunciation or apparent pronunciation errors? How do we handle overlapping utterances, especially when some speakers' voices are not easily recognizable? Recent progress in extensible mark-up languages (XML) is helpful with respect to structured annotation, but guidelines should be carefully designed and documented, and preferably shared among research communities [32.18, 19].

To reach large quantities, imperfect transcription can be used. In the case of broadcast news, closed caption may be useful, although it is not a faithful transcript of the utterances. In this context, lightly supervised training has been investigated [32.20, 21], and reported to be effective in reducing the transcription cost. However, this scheme is workable only when the baseline ASR performance is sufficiently high and reliable.

Another issue is the design of higher-level annotation. In relation to speech recognition, there are many topics in spontaneous speech processing, including sentence/topic boundary detection and key sentence indexing. Since spontaneous speech includes ill-formed phenomena, which are not well explained by conventional linguistic theory, the annotation of these events requires a good deal of analysis and experience us-

ing real data. This often becomes so subjective that agreement among different annotators is not high. Thus, the design of a reliable tag set is itself a topic for research.

32.2.4 Corpus of Spontaneous Japanese (CSJ)

After considering the above issues, we designed and compiled the corpus of spontaneous Japanese (CSJ) [32.22], which is one of the largest spontaneous speech corpora. The CSJ is a primary product of the science and technology agency priority program entitled *Spontaneous Speech: Corpus and Processing Technology*, which was conducted in Japan from 1999 to 2004 [32.5].

The CSJ consists of roughly seven million words with a total speech length of 660 hours. Mainly recorded are monologues of academic presentations (AP) and extemporaneous presentations (EP), as shown in Table 32.2, which amount to about 300 hours each. APs are live recordings made at nine different academic societies covering the fields of engineering, social science, and humanities. EPs are studio recordings of paid laymen's speeches on everyday topics, for example, "the most delightful memory of my life", which were presented in front of a small audience and in a relatively relaxed atmosphere. The topics of APs are related to expert fields, and the vocabularies are often technical, whereas the word distribution in EPs is expected to reflect ordinary Japanese language. While the majority of AP speakers are males aged 20–40 years old, the age and gender of EP speakers are more balanced. In APs, most of the speakers prepared or rehearsed the talk beforehand, but only a few read a manuscript. In summary, AP is real speech in front of a large audience, and EP is more balanced both acoustically and linguistically. The CSJ also includes a number of speech dialogues for the purpose of comparison with monologue speech.

The recordings were manually given both orthographic and phonetic transcription. Phonetic transcription adopts Japanese *kana* characters, but reflects faithful pronunciation rather than orthographic notation. Spontaneous speech-specific phenomena, such as filled pauses, word fragments, reduced articulation and mispronunciation, as well as non-speech events such as laughter and coughing, were also tagged following the carefully designed guidelines. The transcription was very costly, but provides a valuable information resource for statistical modeling of spontaneous speech, including pronunciation variation and disfluency phenomena.

Table 32.2 Contents of the CSJ

Type of speech	#speakers	#talks	Hours
Academic presentations (AP)	838	1006	299.5
Extemporaneous presentations (EP)	580	1715	327.5
Interview on AP	*(10)	10	2.1
Interview on EP	*(16)	16	3.4
Task-oriented dialogue	*(16)	16	3.1
Free dialogue	*(16)	16	3.6
Reading text	*(244)	491	14.1
Reading transcripts	*(16)	16	5.5
		Total	658.8

* counted as AP or EP speakers

One-tenth of the utterances, or half a million words, hereinafter referred to as the *core*, were tagged manually and used for training a morphological analysis and part-of-speech (POS) tagging program [32.23], which was used to automatically analyze the remaining portions of the corpus. The core consists of 70 APs, 107 EPs, 18 dialogues and six read speech files.

These were also tagged with prosodic information as well as higher-level linguistic information. For intonation labeling of spontaneous speech, the traditional J_ToBI was extended to X_JToBI [32.24], in which inventories of tonal events as well as break indices were considerably enriched to reflect spontaneous Japanese.

Linguistic annotation includes the following information:

- dependency structure of *bunsetsu*s, a minimal grammatical unit, consisting of one content word plus adjacent functional words
- clause and sentence boundaries
- discourse boundaries (for part of the core)
- key sentences selected by human subjects
- summaries made by human subjects
- subjective impression of the speech with several adjective-pair dimensions

These high-level annotations are useful for the study of *rich transcription*, and several works using them are explained later herein.

32.3 Analysis for Spontaneous Speech Recognition

Prior to the description of ASR studies, in this section, we present preliminary analyses of spontaneous speech using the CSJ.

32.3.1 Observation in Spectral Analysis

At first, the spectral characteristics of spontaneous speech were analyzed in comparison with those of read speech [32.25]. Utterances from four speech types in the CSJ, which are AP, EP, reading of the transcript of AP (read speech), and dialogues, were used in this analysis. In order to mitigate the effect of individual differences, utterances in different styles by the same five male and five female speakers were compared. Since not only the speakers but also the texts were identical for the reading of the transcribed speeches and the original AP utterances, it was possible to perform a very precise comparative analysis. Each speech had a duration of 10 minutes on average.

It is well known that a significant amount of phones are deleted in spontaneous speech [32.26]. In the CSJ, faithful transcription was manually given considering phone deletions and substitutions, although vowel deletion in consonant–vowel syllables was not taken into account with the transcription using the Japanese *kana* syllable characters.

As an acoustic analysis, 39-dimensional feature vectors, consisting of 12-dimensional mel-frequency cepstral coefficients (MFCC), log-energy, and their first and second derivatives, were extracted from utterances using a 25 ms-length window shifted every 10 ms. The cepstral mean subtraction (CMS) was applied to each utterance. For this experiment, a monophone HMM with three states, each having a single, diagonal Gaussian distribution, was trained for each phone, for each speaker, and for each speaking style. The mean and variance vectors of the 12-dimensional MFCC at the second state of the three-state HMM were extracted for each phone.

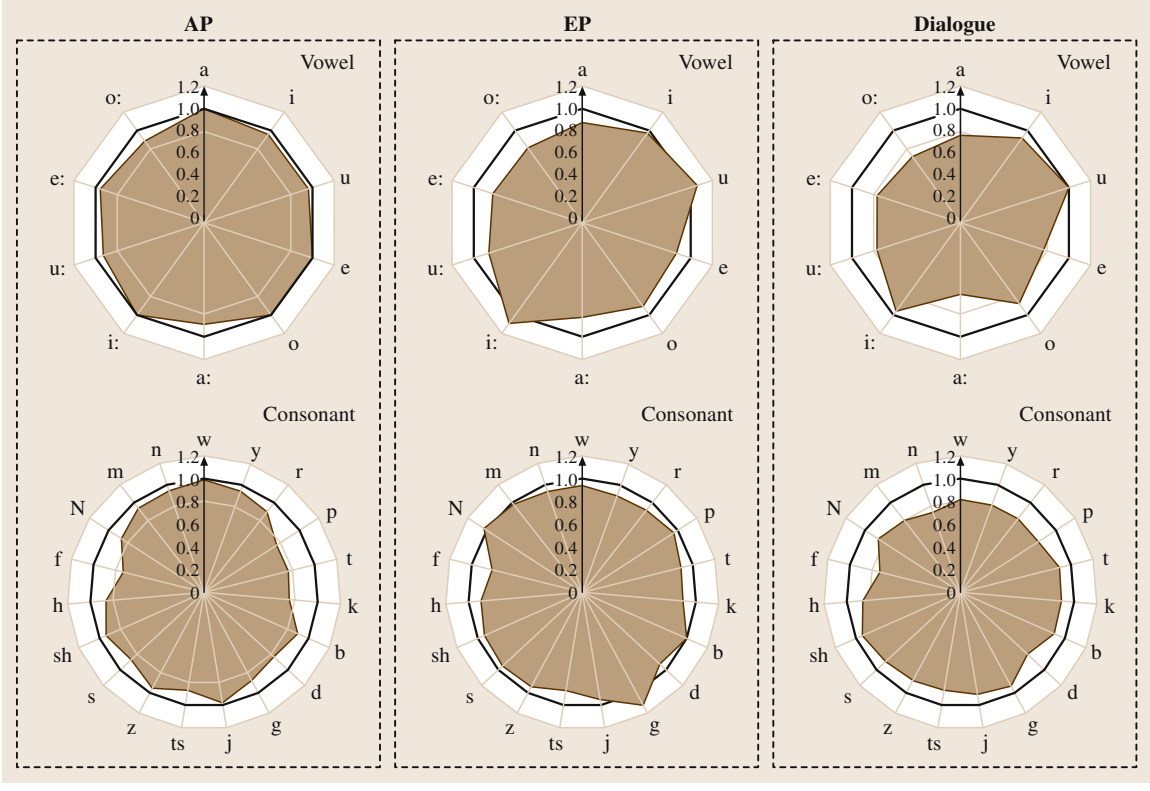


Fig. 32.2 Reduction ratio of the vector norm between each phone and phone center in spontaneous speech to that in read speech

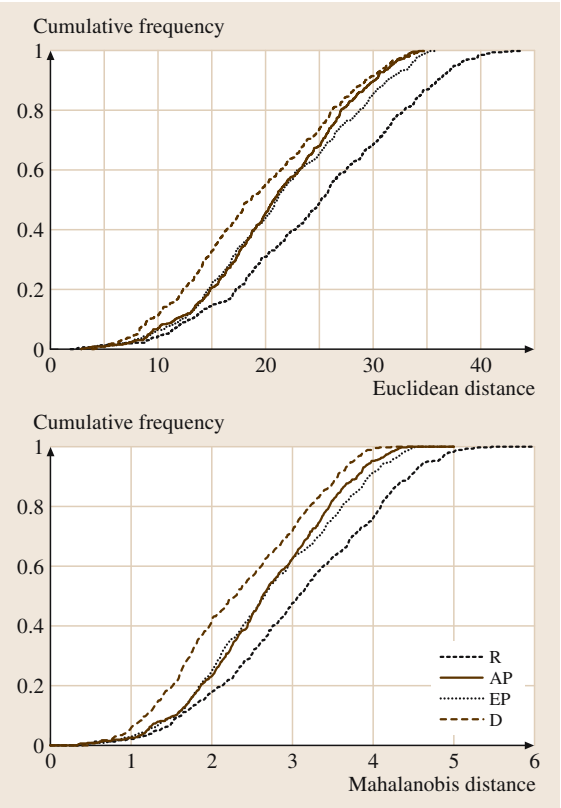


Fig. 32.3 Distribution of distances between phones

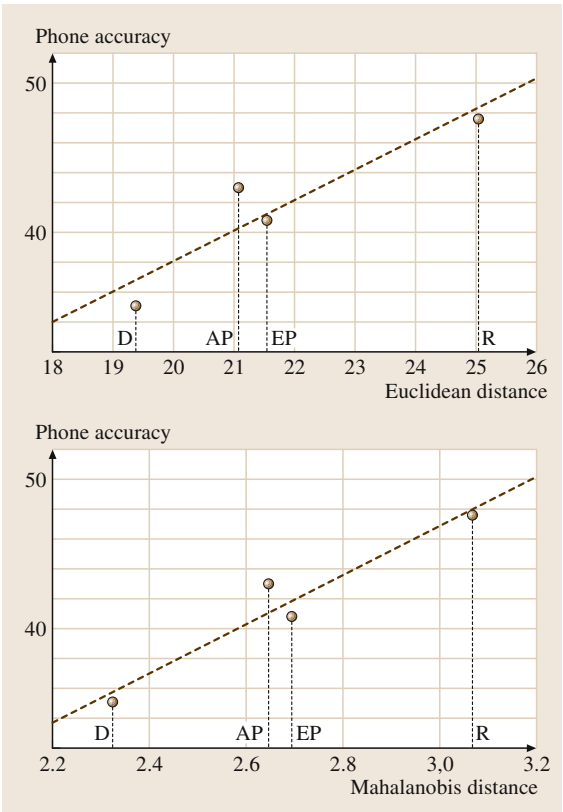


Fig. 32.4 Relationship between phone distances and phone recognition accuracy

In order to analyze the phone reduction quantitatively, Euclidean distances between the mean vector of each phone and the center of the distribution of all phones (the vector averaged over all phones) were calculated, and the ratio of the distance for spontaneous speech (AP, EP, and dialogues) to that of read speech was calculated for each phone. Figure 32.2 shows the reduction ratios averaged over all speakers. The condition of no reduction (the reduction ratio equals to 1) is indicated by a thick line. The reduction of the MFCC space is observed for almost all of the phones in all three speaking styles, and it is most significant for dialogue utterances.

Next, the reduction of the cepstral distance between each phone pair is measured. The Euclidean distance using the mean MFCC vector of each phone and the Mahalanobis distance, which takes into account the variances, were measured. Figure 32.3 shows the cumulative frequency of distances between phones for each speaking style. Here, R and D denote read speech and dialogues, respectively. The distributions are statistically different from each other, except AP and EP. It is shown that the distance between phones decreases as the spontaneity of the utterances increases (D $\gg$ EP $>$ AP $\gg$ R).

Then, the relationship between these distances and ASR accuracy is investigated. Monophone HMMs with a single Gaussian distribution were prepared for each speaking style. A phone network with diphone probabilities was used as a language model for phone recognition. Figure 32.4 shows the relationship between the mean phone distance and the phone recognition accuracy. Correlation coefficients between them are 0.93 in the case of the Euclidean distance and 0.97 for the Mahalanobis distance. The lines in Fig. 32.4 indicate the regression over the four points. These results show that the phone recognition accuracy is strongly correlated with and can be predicted by the mean phone distance.

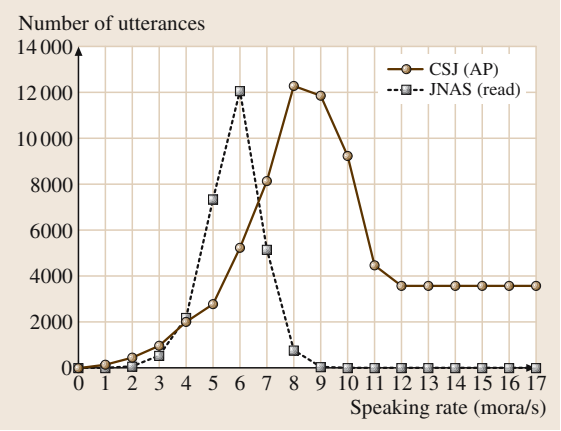


Fig. 32.5 Speaking rate distribution of the CSJ and JNAS corpora

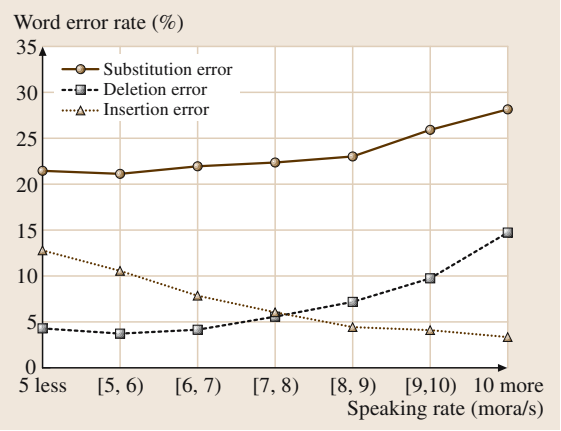


Fig. 32.7 Percentage of substitution, deletion, and insertion errors for each speaking rate

32.3.2 Analysis of Speaking Rate

Next, the analysis results on the speaking rate, which is one of the most prominent characteristics in spontaneous speech, are presented.

In Fig. 32.5, the distributions of the speaking rate in the CSJ (AP of 35 hours) and the JNAS (Japanese newspaper article sentences) corpus (40 hours) are plotted. The speaking rate is estimated for every utterance and is defined as the mora count divided by the utterance duration (sec). For both corpora, manual phonetic transcription was used for defining morae. The mean and standard deviation of the speaking rate of the JNAS corpus were 6.27 and 0.97, and those of the CSJ/AP were 8.70 and 2.10, respectively. This confirms that spontaneous speech is faster than read speech and that the speaking rate variation of spontaneous speech is larger than that of read speech.

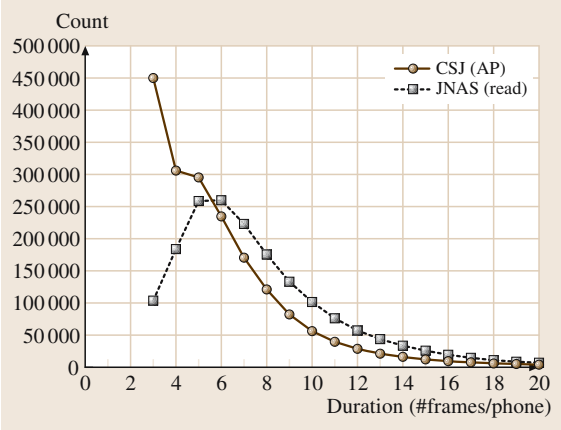


Fig. 32.6 Phone duration distribution of the CSJ and JNAS corpora

The distributions of phone duration in spontaneous and read speech are also plotted in Fig. 32.6. The phone duration is estimated based on the Viterbi algorithm for a given phonetic transcript. Since we used three-state phone HMMs without state skipping, the minimum duration is three frames (30 ms). Many segments in the CSJ may have a shorter duration, but are forcibly aligned to three frames. This may have caused a serious mismatch. Moreover, a fast speaking rate suggests that these segments are poorly articulated and cause problems in ASR.

The relationship between the WER and the speaking rate is also investigated using 15 APs by male speakers. In Fig. 32.7, the breakdown of recognition errors is shown for each speaking rate. Faster utterances are confirmed to generally be more difficult for ASR. Moreover, we observe different tendencies in the errors according to the speaking rate, that is, more deletion errors in faster speech and more insertions in slower speech.

32.3.3 Analysis of Factors Affecting ASR Accuracy

Next, we present a preliminary analysis using a baseline speech recognition system. The acoustic model consists of gender-independent triphone HMMs that have 3000 shared states with 16 Gaussian mixture components. The language model is a standard trigram model of word-pronunciation entries, and the generated lexicon size is approximately 30 000.

Individual differences in ASR accuracy were analyzed using 10 minutes from the CSJ presentations by 51 male speakers [32.27]. Six types of attributes were considered in the analysis: averaged acoustic frame likelihood (AL), speaking rate (SR), word perplexity (PP), out-of-vocabulary rate (OR), filler rate (FR), and repair rate (RR). The speaking rate is defined as the average number of phones per second. The filler rate and the repair rate are the number of filled pauses and self-repairs divided by the number of words, respectively.

The analysis results indicate that the attributes exhibiting a real correlation with the word accuracy are the speaking rate (SR), the out-of-vocabulary rate (OR), and the repair rate (RR). Although the other attributes also have a correlation with the accuracy, the correlation is actually caused through these more fundamentally influential attributes. For example, the word perplexity (PP) is correlated with accuracy, but if its correlation with OR is removed, PP does not correlate well with accuracy.

The following equation was obtained as a result of a linear regression model of the word accuracy (Acc) with the six attributes:

$$\begin{aligned}\mathrm{Acc} = {} & 0.12\mathrm{AL} - 0.88\mathrm{SR} - 0.02\mathrm{PP} \\ & - 2.2\mathrm{OR} + 0.32\mathrm{FR} - 3.0\mathrm{RR} + 95\,. \end{aligned} \tag{32.1}$$

The regression coefficient of RR is −3.0 and that of OR is −2.2. This means that a 1% increase in the repair rate or the out-of-vocabulary rate corresponds, respectively, to a 3.0% or 2.2% decrease in the word accuracy. This is probably because a single recognition error caused by a repair or an out-of-vocabulary word triggers secondary errors due to linguistic constraints. The determination coefficient of the multiple linear regression is 0.48, meaning that roughly half of the variance of the word accuracy can be explained by the model. The normalized representation of the regression analysis, in which the variables are normalized with their means and variances, indicates that the coefficients of SR, OR, and RR are relatively large.

Table 32.3 Statistics of the attributes in the CSJ

	AP	EP
SR: speaking rate (mora/s)	9.05 (1.09)	7.97 (0.79)
PP: perplexity	81.1 (25.3)	82.4 (25.2)
OR: OOV rate (%)	1.45 (0.66)	1.71 (0.82)
FR: filler rate (%)	6.80 (3.44)	5.45 (3.25)
RR: repair rate (%)	1.40 (0.79)	1.25 (0.85)

Upper row: mean, lower row: standard deviation

The statistics of these attributes are listed in Table 32.3 for the AP and EP parts of the CSJ. Here, the speaking rate (SR) is computed by dividing the number of morae by the total duration (sec) of the talk, excluding pauses. Note that, in this subsection, the speaking rate is computed as an average over the entire talk, whereas it was computed utterance by utterance in Sect. 32.3.2 for more precise analysis. This difference in the definition greatly influences the standard deviation. In Table 32.3, it is observed that speaking in AP is faster and more disfluent; more specifically it has more fillers and repairs. For reference, in the JNAS read speech corpus, the speaking rate is slower (7.36 mora/sec) and there are no fillers and repairs.

Based on these analyses, test sets for ASR evaluation were prepared for the AP and EP parts of the CSJ. These sets are designed by balancing three factors of speaking rate (SR), perplexity (PP), and repair rate (RR) that affect the ASR performance. Here, we use the perplexity (PP) instead of the out-of-vocabulary rate (OR), because OR intrinsically depends on vocabulary and easily varies when the lexicon is modified.

32.4 Approaches to Spontaneous Speech Recognition

Following the observation of several characteristics in spontaneous speech, in this section, major approaches to speech recognition are reviewed with respect to acoustic and language models.

32.4.1 Effect of Corpus Size

First, the effect of the corpus size was investigated using the CSJ. Fig. 32.8 shows the WER, adjusted test-set

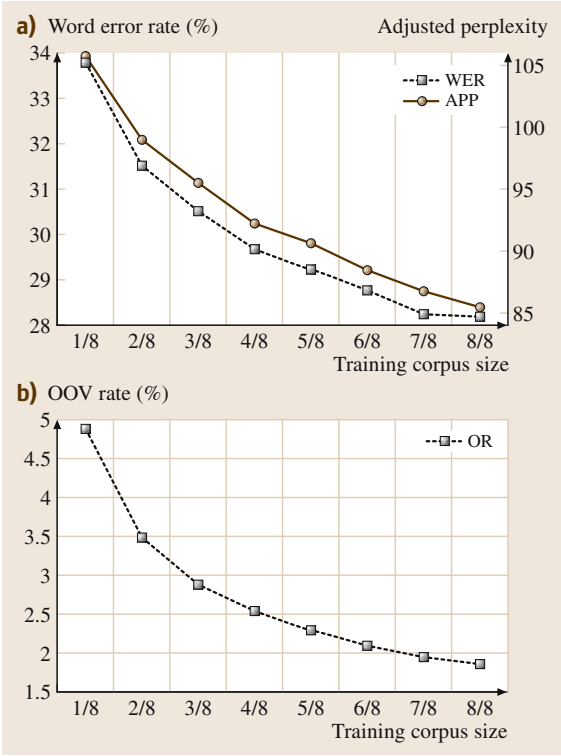


Fig. 32.8 **(a)** WER, adjusted test-set perplexity (APP); **(b)** and out-of-vocabulary rate (OR) as a function of the size of language model training data

perplexity (APP), and out-of-vocabulary rate (OR) as a function of the size of the language model training data, with the condition that the acoustic model is fixed. The adjusted perplexity was used to normalize the effect of the increase in the vocabulary size on the perplexity according to the increase in the training data size. By increasing the language model training data size from 1/8 (0.86M words) to 8/8 (6.84M words), the WER, perplexity and OOV rate are reduced by 17%, 19%, and 62%, respectively.

On the other hand, Fig. 32.9 shows the WER as a function of the size of the acoustic model training data, when the language model is made using the entire training data set. By increasing the acoustic model training data from 1/8 (68 hours) to 8/8 (510 hours), the WER is reduced by 6.3%.

These results show that the WER is significantly reduced by the increase of the training data. It is confirmed that the size of the CSJ has a strong impact on modeling spontaneous speech. Comparing Fig. 32.8 and Fig. 32.9, the effect of WER reduction on the language

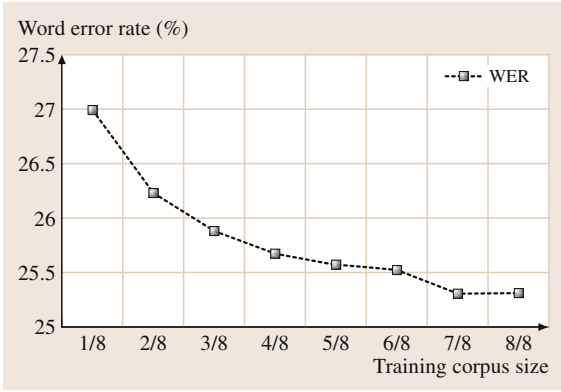


Fig. 32.9 WER as a function of the size of acoustic model training data

model is much greater and appears not to saturate even when using the entire corpus (rightmost plot). This suggests the necessity for more language model training data.

32.4.2 Acoustic Modeling

As shown in the previous section, spontaneous speech has greater variation both in spectral and temporal structures than read-style speech. As the acoustic variation is largely dependent on speakers, feature normalization techniques such as vocal-tract-length normalization (VTLN) [32.28, 29] are effective. The speaker-adaptive training (SAT) [32.30, 31] scheme has also been extensively studied.

On the other hand, it should be questioned whether the conventional HMM is appropriate for modeling spontaneous speech. In principle, an HMM assumes piecewise transitions of states with respective distributions, even though it can incorporate dynamic spectral features such as ΔCepstrum. The segment model [32.32] and the trajectory model [32.33] have been proposed to capture the dynamics of speech. These models, however, have huge computational costs and often require the estimation of segmental boundaries. Thus, they are usually applied only in the rescoring stage after initial HMM-based decoding, and their potential may not have been fully exploited.

Another approach is to model syllables or functional words, instead of phones, in order to model several variants that are characteristic of frequent words such as 'that'. These types of models are often effective in context-independent models, but are difficult to extend to context-dependent models, which have enor-

mous numbers of combinations and encounter the data sparseness problem.

Many factors affect acoustic variation in spontaneous speech. A machine learning approach is to list and include these factors in the decision tree-based state clustering in context-dependent phone HMM. In this case, the context is extended from conventional phonetic context to any factors causing variations such as speaking rate [32.34], whether or not the phone concerned is included in a functional word, and whether or not the phone is stressed. This approach requires labeling of these kinds of information, which would be expensive if performed manually and unreliable if performed automatically. Moreover, the binary decision in the decision tree may not be appropriate, because the effects of these factors are not so apparent.

A more-generalized framework is dynamic Bayesian network (DBN) modeling [32.35] or graphical modeling [32.36,37], which includes these factors in the topology of an HMM-like network and estimates their influence in a probabilistic manner. One example that counts speaking rate is presented in the following subsection.

32.4.3 Models Considering Speaking Rate

As described in Sects. 32.3.2 and 32.3.3, one of the most important issues in spontaneous speech recognition is how to cope with the speaking rate fluctuation, especially fast speaking. As such, a number of studies have been conducted on this topic.

Initially, dedicated acoustic modeling to fast speech, where fast-speech models are prepared in parallel with the normal models, was investigated (e.g., [32.38–40]). However, this explicit approach is not adequate because it is not straightforward to classify the training data, especially frame by frame, to fast and poorly articulated samples.

Several studies have proposed dedicated acoustic analysis [32.41–43] for fast speech. These methods estimate the phone boundaries or speaking rates, and normalize the speaking rate by changing the analysis frame or acoustic model according to the estimated speaking rate.

Okuda et al. [32.44] proposed a speaking rate compensation method, which decodes an input utterance using several sets of frame period and frame length parameters for speech analysis. The method, then, selects the set with the highest likelihood normalized by the frame period. Furthermore, this approach was applied to the training phase of the acoustic model just like VTLN-based training.

Shinozaki et al. [32.45] proposed an acoustic model that adjusts mixture weights and transition probabilities of an HMM for each frame according to the local speaking rate. The proposed model, implemented based on a dynamic Bayesian network framework, has a hidden variable representing the variation of the *mode* of the speaking rate, and its value controls the parameters of the underlying HMM.

Nanjo et al. [32.46] proposed a hybrid approach, called speaking-rate-dependent decoding, which applies the most adequate acoustic analysis, phone models and decoding parameters according to the estimated speaking rate. Several methods were investigated and their selective application led to improved accuracy.

32.4.4 Pronunciation Variation Modeling

The phonetic variation caused by spontaneous utterance can also be modeled in a pronunciation dictionary, in which a list of possible phone sequences for each word is defined. While the orthodox pronunciation forms are referred to as baseforms, the variants observed in spontaneous speech are called *surface forms*. Several studies have also addressed a framework that jointly optimizes pronunciation entries and acoustic models [32.47], or the model unit itself [32.48].

These surface form entries are often derived from speech data by aligning them with phone models [32.26, 49]. In the CSJ, actual phonetic (*kana*) transcription is given manually, so the set of surface forms is easily defined. However, the simple addition of surface form entries results in the side-effect of false matching. Thus, effective but constrained use of these surface forms are necessary. One approach is context-dependent modeling, for example, dedicated surface forms are chosen according to the speaking rate [32.38, 50, 51].

Another approach is statistical modeling, which is similar to language modeling. Namely, the unigram probability of each pronunciation form is assigned and is multiplied by the language model probability in decoding [32.52–54]. In this case, the statistical framework of ASR is reformulated as:

$$w' = \arg\max_{w,p} P(x|p)P(p|w)P(w)\,. \qquad (32.2)$$

Here, $P(p|w)$ is the pronunciation probability of surface form p for word w. *Nanjo* et al. [32.46] investigated the comparison of statistical models using the CSJ, and concluded that cutting off less frequent surface forms is crucial, and that the unigram model is effective, whereas the trigram model offered a marginal benefit.

When the surface form is derived for word units, it is dependent on the task and corpus and is not necessarily applicable to different tasks. Phone-based modeling of pronunciation variation is more general and portable to various lexicons. Surface forms are obtained by applying such a model to phone sequences of baseforms. As the modeling framework, the decision tree [32.55], the neural network [32.56], and the confusion matrix [32.57] have been proposed.

Akita et al. [32.58] proposed generalized modeling of subword-based mapping between baseforms and surface forms using variable-length phone context. The variation patterns of phone sequences are automatically extracted together with their contexts of up to two preceding and following phones, which are determined by their occurrence statistics. A set of rewrite rules are then derived with their probabilities and variable-length phone contexts. The model effectively predicts pronunciation variations depending on the phone context using a back-off scheme.

32.4.5 Language Model

Language model training for spontaneous speech is much more difficult than that for dictation systems, which can make use of huge language resources such as newspaper articles and Web pages. Most of the available language data are written text, and are mismatched with the spoken style. For language modeling of spontaneous speech, a great deal of transcription is essential, but has a huge cost.

Even the CSJ with a text size of 7M words, which is one of the largest spontaneous speech corpora, is comparable to or smaller than four months' worth of newspaper articles with respect to the text size. As shown in Fig. 32.8, increases in the amount of training data have a significant effect on ASR accuracy, and the improvement is not saturated even with the full CSJ.

Another problem in language modeling is how to deal with disfluency in spontaneous speech. Initially, insertion of fillers was assumed as an interruption of the n-gram chains, and so was often disregarded in n-gram training or prediction. Such a strategy may be effective when the disfluency is regarded as exceptional, as in dictation systems or when the size of the training corpus is small. When large corpora of spontaneous speech were built, however, the occurrence of fillers was confirmed to have such a regularity that the conventional n-gram model dealing with them as ordinary words was the most effective. Similarly, long pauses can be modeled as individual lexical entries in the n-gram model and are useful for improving the prediction of words.

The most widely used solution to enhance language model training data is to combine or interpolate with other existing text databases that are not necessarily spontaneous speech corpora but are related to the target task domain. These include proceedings of lectures, minutes of meetings, and closed captions for broadcast programs. Recently, the World Wide Web has become a major language resource [32.59], and not a few Web sites contain spoken-style documents, such as records of lectures and meetings. Several studies have addressed effective query generation to collect relevant Web pages [32.60, 61] and the selection of spoken-style texts [32.62].

Akita et al. [32.63] proposed a *translation* approach that estimates language model statistics (n-gram counts) of spontaneous speech from a document-style large corpus based on the framework of statistical machine translation (SMT). The translation is designed for modeling characteristic linguistic phenomena in spontaneous speech, such as insertion of fillers, and estimating their occurrence probabilities. These contextual patterns and probabilities are derived from a small parallel aligned corpus of faithful transcripts and their documented records. This method was successfully applied to the estimation of the language model for National Congress meetings from their minute archives.

32.4.6 Adaptation of Acoustic Model

Since the variation of acoustic features is very large in spontaneous speech, speaker adaptation of the acoustic model is effective and is almost essential. Although many factors affect the acoustic characteristics of spontaneous speech such as speaking rate and speaking styles, speaker adaptation is a simple solution to handle all of these factors in an implicit manner. The acoustic model adaptation also involves channel adaptation, that is, the characteristics of rooms and microphones are also normalized.

In particular, in lectures or meetings, each speaker makes many utterances in the same session. Thus, a considerable amount of data is available to conduct unsupervised adaptation in a batch mode, where the initial ASR result with the speaker-independent model is used for adaptation of the acoustic model, which is then used for rescoring or re-decoding. Standard adaptation techniques such as maximum-likelihood linear regression (MLLR) are used, and filtering the reliable

ASR hypotheses with confidence measures can also be incorporated.

32.4.7 Adaptation of Language Model

Adaptation of the language model is also important to deal with a variety of topics and speaking styles. In lectures and meetings, the topic is focused and consistent throughout the entire session. Therefore, language model adaptation is feasible even in an unsupervised or batch mode, as in the acoustic model adaptation, and computationally expensive methods can be allowed in offline transcription tasks.

The simplest methods are the cache model [32.64, 65], and the trigger model [32.66], which weigh the probability of words recently used in the utterances or talk, or those directly related to the previous topic words. *Troncoso* et al. [32.67] proposed a trigger-based language model adaptation method aimed at meeting transcription. The initial ASR result is used to extract task-dependent trigger pairs and to estimate their statistics. This method achieved a remarkable perplexity reduction of 28%. However, these cache or trigger models do not necessarily bring about a significant improvement in ASR accuracy because they were not robust against initial ASR errors, i. e., errors can propagate easily. Moreover, the methods are usually only applicable to the rescoring framework and are constrained by the quality of initial word graphs.

Thus, a potentially more-effective scheme is to adapt or reconfigure n-gram models for re-decoding, as in the acoustic model adaptation. *Yokoyama* et al. [32.68] constructed an n-gram model from the initial ASR result and interpolated it with the baseline model. They incorporated a class-based model for robust estimation from the small and erroneous ASR text [32.69]. *Lussier* et al. [32.70] extended the framework to optimize the interpolation weight using the EM (expectation-maximization) algorithm. *Nanjo* et al. [32.46] investigated methods to select the most relevant texts from the corpus [32.71] based on the initial ASR result. As a criterion for text selection, they used the tf-idf (term frequency and inverse document frequency) measure and perplexity by the n-gram model generated from the initial ASR result, and demonstrated that they have comparable and significant effects in reducing the WER. The results for the CSJ AP test set are summarized in Table 32.4.

Another approach is to model topics in an implicit manner, because the topics are not so definite and often become complex. One implementation of this concept is a mixture of multiple language models covering various topics [32.72]. Adaptation based on interpolation can be performed by weighting or emphasizing models relevant to the input speech.

Recently, latent semantic analysis (LSA), which maps documents into implicit topic subspaces using singular value decomposition (SVD), has been investigated extensively for language modeling [32.73]. A probabilistic formulation, PLSA (probabilistic latent semantic analysis) [32.74], is powerful for characterizing the topics and documents in a probabilistic space and predicting word probabilities. *Akita* et al. [32.75] proposed an adaptation method based on two subspaces of topics and speaker characteristics. Here, PLSA was performed on the initial ASR result to provide unigram probabilities conditioned on the input speech, and the baseline model is adapted by scaling n-gram probabilities with these unigram probabilities. The method was applied to automatic transcription of panel discussions and was shown to be effective in reducing both perplexity and WER.

A summary of the current ASR performance for the CSJ AP test set, using the baseline model enhanced with the complete corpus and a larger Gaussian mixture (160K in total), is given in Table 32.5. By combining the adaptation of acoustic and language models using the initial ASR result, the word accuracy was improved to 83%.

Table 32.4 The effect of language model adaptation

Method	WER	PP
Baseline*	30.5	74.9
Text selection by perplexity	29.7	68.7
Text selection by tf-idf	29.1	70.2
n-gram made from ASR result	28.8	51.8
Combination of all of the above	27.6	46.7

* Baseline trained with an intermediate version of the CSJ

Table 32.5 ASR results for CSJ/AP (academic presentations)

Method	WER
Baseline	22.2
+ MPE (minimum phone error) training	19.9
+ AM adaptation (MLLR)	18.1
+ LM adaptation (n-gram from ASR)	17.1

32.5 Metadata and Structure Extraction of Spontaneous Speech

The transcript of spontaneous speech is not easy to read, even if it contains no recognition errors. The primary factor is disfluency, such as fillers, repetitions, self-repairs, and word fragments. These make spontaneous speech transcripts very different from written text. Actually, in the official meeting records of the Japanese Congress, we observe that approximately 11% of words are changed from the verbatim transcript (deletion 8.5%, insertion 1.0%, substitution 1.8%). Although there are a number of colloquial expressions corrected to more formal expressions, the great majority of the differences are caused by disfluencies. This means that even if we realize a perfect ASR system in the conventional criteria where speech is faithfully decoded into uttered word sequences, it has an edit distance (WER) of more than 10% from the text made by a human transcriber. This phenomenon is observed in any language. The ultimate intelligent transcription system should be able to clean these disfluencies automatically, but the first step is to model and detect these phenomena.

The second factor is segmentation of sentence and paragraph units, which is essential in written text. It seems that in most written languages the unit of the sentence is well defined, and thus can be objectively annotated, even though the end-of-sentence marks (the period in English) may differ. On the other hand, units smaller than the sentence (often marked by commas in English) and units larger than the sentence (marked by line breaks and indentation) are often subjective. Therefore, we focus on the sentence unit. Conventionally, punctuation marks are not counted when measuring ASR accuracy, and many ASR systems do not output these marks. In some dictation systems, users have to utter these marks explicitly, which is never expected in spontaneous speech. Thus, automatic segmentation of transcripts into sentences and paragraphs is another key factor in developing a *rich transcription* system.

Several studies in this direction have been performed for switchboard conversations [32.76]. In rich transcription (RT) evaluation of the DARPA EARS program, the metadata extraction (MDE) framework was designed for the following four tasks [32.77, 78].

- sentence unit (SU) detection
- edit word detection (repair detection)
- filler word detection
- interruption (disfluent) point (IP) detection

Here, the interruption point detection is, by definition, closely related to the detection of edit words and fillers.

Similarly in the CSJ, the following tags are given to the core of 0.5M words [32.22].

- sentence and clause boundaries
- filler tag (F tag)
- disfluency tag (D tag) for repairs and repetitions

We introduce a number of the key methods and results in the following subsections.

32.5.1 Sentence Boundary Detection

Detection of the sentence unit is vital for linguistic processing of spontaneous speech, since most of the conventional natural language processing (NLP) systems assume that the input is segmented by sentence units. Sentence segmentation is also an essential step to key sentence indexing and summary generation, which are described in the next section.

In spontaneous speech, especially in Japanese, in which subjects and verbs can be omitted, the unit of the sentence is not so evident. In the CSJ, therefore, the clause unit is first defined based on the morphological information of end-of-sentence or end-of-clause expressions. The sentence unit is then annotated by human judgment considering syntactic and semantic information.

Several approaches to automatic detection of sentence boundaries are described in the following subsections.

Statistical Language Model (SLM)

In fluent speech or read speech of well-formed sentences, it is possible to assume that long pauses can be interpreted as punctuation marks, and the insertion of periods (sentence boundaries) or commas can be determined by the neighboring word contexts.

Thus, the baseline method makes use of an n-gram statistical language model (SLM) that is trained using a text with punctuation symbols, in order to determine a pause to be converted to a period. Specifically, for a word sequence around a pause, $X = (w_{-2}, w_{-1}, \textit{pause}, w_1, w_2)$, a period is inserted at the place of the pause if $P(W_1) = P(w_{-2}, w_{-1}, \text{period}, w_1, w_2)$ is larger than $P(W_2) = P(w_{-2}, w_{-1}, w_1, w_2)$ by some margin. Actually, this decoding is formulated as the maximization of a likelihood $\log P(W) + \beta \cdot |W|$, where $|W|$ denotes the number of words in W and β is the insertion penalty widely used in ASR.

In spontaneous speech, however, approaches that rely heavily on the pauses are not successful. Speakers put pauses in places other than the ends of sentences for certain discourse effects, and disfluency causes irregular pauses (interruption points), while consecutive sentences are often continuously uttered without a pause between them.

Support Vector Machines (SVMs)

A simpler but more general approach is to treat the pause duration as one of the features in addition to the lexical features, and feed them into a machine learning framework. We adopted support vector machines (SVM) because there are a wide variety of cue expressions suggesting sentence endings in Japanese. In this case, sentence boundary detection is regarded as a text chunking problem [32.79], and we adopt the IE labeling scheme, where I and E denote the inside chunk and end of chunk, respectively. For every input word, a feature vector is composed of the preceding and the following three words, together with their POS tags and the durations of the subsequent pauses, if any. The pause duration is normalized by the average in a turn or a talk, because it is affected by the speaking rate and significantly different between speakers. Dynamic features or estimated results of preceding input parts can also be fed into SVM. The SVM is considered to be powerful for handling a very large number of features and finding the critical features called *support vectors*.

Maximum-Entropy Method (MaxEnt) and Conditional Random Fields (CRF)

The maximum-entropy method (MaxEnt) [32.80, 81] is also widely used in NLP to combine multiple features F in an exponential form and classify event e given F:

$$p(e|F) = \frac{1}{Z(F)} \prod_i \mathrm{e}^{\lambda_i f_i(e,F)} , \qquad (32.3)$$

where $f_i(e, F)$ is binary depending on the match of the i-th feature in the current data F, and the weights λ_i is trained to maximize $p(e|F)$ for training data. Similar features used for SVM can be fed into the model, although real-valued features such as pause duration are usually quantized into several bins.

Conditional random fields (CRFs) [32.82] are an extension of MaxEnt to handle sequential inputs. It is also an extension of HMM [32.83] so as to flexibly handle multiple features which may not be independent, and to be optimized for classification. Thus, sentence boundary detection is formulated as a chunking problem using the IOE labeling scheme or a similar scheme.

Unlike SVM, MaxEnt and CRF are statistical models, and they compute a posterior probability, which can be used as a confidence measure. This property is useful when we want to control the operating point of recall and precision, or in case that the result is combined with other processing using higher-level knowledge sources. On the other hand, SVM does not estimate probability distribution but focuses on classification boundaries, thus can be more robustly trained with sparse data of a huge dimension of features.

Experimental Evaluations

First, we present the results evaluated in the CSJ [32.84]. The test set was that used for ASR evaluation and consists of 30 presentations or 71 000 words in total. Both SLM and SVM described in the previous subsections were trained with the core 168 presentations of 424 000 words, excluding the test set. In this experiment, we used ASR results without conducting speaker adaptation and the WER was approximately 30%. The results are summarized in Table 32.6, where the recall, precision, and F-measure are computed for sentence boundaries. Correct transcripts (text) are used for reference. SVM realizes higher performance in the text case, but significantly degraded it in the ASR case. On the other hand, SLM shows robustness for erroneous input. It is noteworthy that performance degradation by using ASR is much smaller than the WER.

In [32.85], *Liu* et al. reported the sentence boundary detection results in MDE tasks of rich transcription evaluation (RT-04). They trained HMM, MaxEnt, and CRF methods, which gave similar performance, and combined them by obtaining their majority vote for further improvement [32.86]. They used the boundary detection error rate, which counts deletion (miss) and insertion (false detection) errors. It is comparable to the summation of the complements of recall and precision if the number of detected boundaries is the same

Table 32.6 Results of sentence unit (boundary) detection in the CSJ

	Recall	Prec.	F-measure
SLM (text)	79.2	84.6	81.8
SLM (ASR)	70.2	71.6	70.9
SVM (text)	82.7	88.0	85.3
SVM (ASR)	56.4	66.0	60.9

Table 32.7 Summary of sentence unit detection

	CTS	BN	CSJ
Training data size	484K	353K	424K
Test-set size	35K	46K	71K
WER of ASR	14.9	11.7	30.2*
SU detection error (text)	26.4	48.2	35.2
SU detection error (ASR)	36.3	57.2	57.7

* ASR result for the CSJ is not latest

as that of the correct boundaries. A summary comparison of the sentence unit detection for conversational telephone speech (CTS) and broadcast news (BN) is shown in Table 32.7, in which the results for the CSJ are also given for comparison. The performance for CTS is much better than for BN, because BN has more complex sentences, while in CTS, pronouns and back-channels work as good predictors for boundaries between shorter sentences. Moreover, speaker turn information, if available, will greatly help the system identify the end of sentences. Compared to these results, the performance for the CSJ monologue is not lower, despite the larger WER.

Liu et al. claimed that prosodic features are crucial to performance, working robustly against ASR errors. On the other hand, some works [32.87] have suggested that the most dominant prosodic feature is pause information, which is primarily used in the CSJ evaluation. Pitch and energy are apparently useful in the perception of sentence boundaries, but they are not easy to parameterize reliably. A typical solution is to introduce a separate classifier based on a decision tree or neural network that computes the likelihood integrating possible prosodic features [32.86, 88].

Another approach for further improvement is to incorporate higher-level linguistic information, such as syntactic dependency and caseframe structures. *Shitaoka* et al. [32.89] presented an interactive framework of parsing and sentence boundary detection, and showed that dependency structure analysis can help sentence boundary detection and vice versa. *Hamabe* et al. [32.90] addressed the detection of quotations, which is similar to sentence boundary detection, but involves analysis of very complex sentence structures.

32.5.2 Disfluency Detection

Disfluency is another prominent characteristic of spontaneous speech. Disfluency is inevitable because humans make utterances while thinking about what to say, and the pipeline processing is often clogged. Thus, the detection of disfluencies may be useful for analyzing the discourse structure or speaker's mental status. However, disfluencies should be removed for improving readability and applying conventional NLP systems including machine translation and summarization.

Disfluency is classified into the following two broad categories:

- fillers (such as 'um' and 'uh'), including discourse markers (such as 'well' and 'you know'), with which speakers try to fill pauses while thinking or to attract the attention of listeners;
- repairs, including repetitions and restarts, where speakers try to correct, modify, or abort earlier statements.

Note that fillers usually appear in the middle of repairs.

In the CSJ, specific tags are given for respective categories. The filler rate and the repair rate (ratios against the number of words) are listed in Table 32.3. Here, APs have more disfluencies because presenters speak much faster. In the RT-04 data, the filler rate is 9.2% for CTS and 2.1% for BN [32.85]. BN has fewer fillers because it is delivered mostly by professional anchors.

Lexical filler words are usually obtained as the output of the ASR system, and their recognition accuracy is much the same as that of ordinary words. However, there are a number of words that also functions as non-fillers such as 'well' in English and '*ano*' in Japanese. For these distinctions, prosodic features will be useful since we can recognize fillers even for unfamiliar languages. *Quimbo* et al. [32.91] investigated the difference in prosodic features in these words in Japanese.

On the other hand, the detection of self-repairs involves much more complex processes. The most conventional approach is to assume the repair interval model (RIM) [32.92], which consists of the following three parts in order: the reparandum (RPD), the portion to be prepared; the disfluency (DF), fillers or discourse markers; and the repair (RP), the portion to correct or replace the RPD, as shown below:

(RPD) ! (DF) (RP)
(ex.) "I'm going {RPD: to Tokyo} ! {DF: no} {RP: to Kyoto}"

The first step to the self-repair analysis based on this model is to detect the DF or interruption points (IP), denoted with '!' in the above, which seem rela-

Table 32.8 Example of presumed discourse markers derived from academic presentations

tsugi (next), *sakihodo* (previously), *ima* (now), *jiQsai* (actually), *koNkai* (this time), *saigo* (lastly), *saisho* (firstly)
keNkyuu (study), *setsumei* (to explain), *haQpyou* (to present)
keQka (result), *jiQkeN* (experiment), *moNdai* (problem), *hyouka* (evaluation), *houhou* (method)
Listed in Japanese with corresponding English translations

tively easy to spot. The DF usually consists of filler words, and IP detection can be formulated in much the same manner as the sentence boundary detection using neighboring lexical features together with prosodic features [32.93]. *Nakatani* et al. [32.92] pointed out the importance of prosodic features in this process. In rich transcription evaluation (RT-04), the patterns following the RIM are mainly focused, and the aforementioned approach was taken by *Liu* et al. [32.93]. For this purpose, they used the same classifiers based on HMM, MaxEnt, and CRF. After finding the DF or IP, the next step is to identify the starting point of the reparandum (RPD). Heuristic rule sets could be handcrafted for this purpose. A more elaborate statistical model was introduced to make alignment between the reparandum (RPD) and repair (RP) segments in [32.94]. As an even more general model, *Honal* et al. [32.95] introduced a statistical machine translation framework, which models self-repairs being caused through a noisy channel. These models work effectively if fillers are present in the DF position (following IP), and parts of RPD and RP segments are lexically matched, for example, 'on Monday, no, on Tuesday.' The latter assumption generally holds in English, in which phrases start with function words or pronouns and speakers repeat them in self-repairs. In this case, the removal of the RPD and DF is sufficient to clean the transcript.

In the CSJ or Japanese monologue, however, we observe many cases that do not satisfy this assumption or RIM. First, DF or filler words are often absent. Second, RPD and RP segments often have nothing in common on the surface level, although they may be semantically related, for example, "*ana* (hole) ! *mizo* (trench) *wa* . . . " This phenomenon makes it extremely difficult to perform machine learning using lexical features. Actually, using SVM, we obtained a detection accuracy (F-measure) of 77.1% for the cases in which the RPD and RP segments have some words in common, but only 20.0% otherwise. This result suggests that high-level semantic information is necessary for further improvement.

32.5.3 Detection of Topic and Discourse Boundaries

Topic and discourse boundary detection of spontaneous speech such as broadcast news and lectures has also been studied. It is formalized as a topic detection and tracking (TDT) task, where broadcast news is the major target and text-based techniques are applied and extended to ASR results. Incorporation of prosodic features has also been investigated. *Passonneau* and *Litman* [32.96] addressed discourse segmentation of narrative monologues using various combinations of prosodic features, cue phrase features, and noun-phrase reference features. They crafted decision rules combining these features both by hand and by machine learning. *Shriberg* et al. [32.83] applied machine learning techniques of decision tree learning and HMM, which were also used in the sentence boundary detection, to topic segmentation of broadcast news. *Haase* et al. [32.97] also tried discourse segmentation of monologue news reports using several prosodic features. *Kawahara* et al. [32.98] presented a method that automatically extracts discourse markers useful for the detection of topic boundaries in oral presentations of the CSJ. The presumed discourse markers were derived in a totally unsupervised manner based on word statistics and pause information. An example list of discourse markers derived from academic presentations (APs) is given in Table 32.8.

In dialogues or meetings involving multiple talkers, the detection of speaker turns is also critical. While the approach based on speaker identification techniques is commonly used, the use of linguistic information such as discourse information should be explored. Recently, speaker identification research itself is being extended to incorporate linguistic information.

32.6 Speech Summarization

As described in Sects. 32.3 and 32.4, ASR accuracy for spontaneous speech is not yet sufficient for quality automatic transcription. Recognition errors result in transcripts with irrelevant or incorrect information. In addition, spontaneous speech is ill-formed and usually includes redundant information caused by disfluencies. Direct transcription is therefore not always useful, and processes for extracting important information and removing irrelevant information are necessary for effective presentation of the transcript.

Automatic speech summarization is an approach aimed at accomplishing this goal. By condensing the important points of long audio materials or speech documents, such as broadcast programs and lectures, and presenting them in a concise form, the system will reduce the time needed for reviewing them and improve the effectiveness of their retrieval. Thus, speech summarization technology is expected to play an important role in building various speech archives, by providing a valuable means for efficiently accessing and absorbing more information in a much shorter time. It will also contribute to the advancement of question answering (QA) systems targeting speech documents, because both involve extracting the most salient information from such documents.

Speech summarization poses a number of significant challenges that distinguish it from general text summarization [32.99]. Applying text-based technologies [32.100] to speech is not always viable and systems are often not equipped to capture speech-specific phenomena. One fundamental problem with speech summarization is that it must deal with ASR errors and disfluencies [32.101, 102]. Another problem is that speech has no clear structure, whereas text contains not only sentences and paragraphs defined by punctuation and line breaking, but also titles for articles and individual sections, which provide useful information for summarization. These issues were addressed in the previous section, but automatic extraction of these kinds of information is inevitably error-prone.

32.6.1 Categories of Speech Summarization

Speech summarization can be classified into the following two categories based on its presentation form:

- speech-to-text summarization (output by text)
- speech-to-speech summarization (output by speech)

Speech-to-text summarization has advantages in that: (1) the documents can be easily looked through, (2) the parts of the documents that are of interest to the user can be easily spotted, and (3) information extraction and retrieval techniques can be easily applied to the documents. However, it also has disadvantages in that incorrect information caused by ASR errors is inevitable, and prosodic information such as nuance and emotion of the speaker, which is conveyed only in speech, cannot be presented. On the other hand, speech-to-speech summarization preserves the acoustic information included in the original speech.

Current approaches to speech summarization can be classified into two categories:

- sentence extraction (extracting important sentences or clauses as they are, and concatenating them in order)
- sentence compaction (generating summarized sentences by shortening, fusing, or even modifying the transcript)

The manual summarization process with text output by human editors corresponds to the latter, and the former can also be used as an intermediate step to the latter [32.103]. By applying sentence compaction, flexible output can be generated by concatenating key information and removing redundant expressions in speech. Users can then quickly browse and understand the content. Therefore, the sentence compaction approach is appropriate for speech-to-text summarization, and this method can be applied to any size of speech documents from headline news to long lectures.

In the case of speech-to-speech summarization, the extraction approach has an advantage in that the output can be easily made by concatenating corresponding segments from the original speech. On the other hand, the sentence compaction approach requires a text-to-speech (TTS) system, because concatenation would produce unnatural sounds with clipping noises and cannot be applied to units smaller than words. However, the quality of the synthesized speech, even by state-of-the-art TTS systems, is not as high as natural speech. In addition, the TTS-based method cannot mitigate the problems which inevitably result from ASR errors. Therefore, sentence extraction is more appropriate for speech-to-speech summarization, even though the reproduced speech includes redundant expressions and disfluencies. By indexing sentences, the system can provide users

with a useful method to access relevant audio segments quickly.

In the following subsections, a brief overview of the respective approaches is provided. For each of the CSJ core and ASR test sets (199 presentations in total), three subjects were asked to construct summaries according to the following guidelines, where the summarization ratio corresponds to the ratio of the length of the summary to that of the original text:

- a set of key sentences extracted from the transcript (50% and 10% summarization ratios in terms of the number of sentences)
- a summary generated by editing the transcript (50% and 10% summarization ratios in terms of the number of characters)

For the former, the subjects were instructed to extract important sentences, first choosing a set of 50% of the sentences in a talk, and then choosing a subset thereof to obtain an overall 10% summarization ratio.

32.6.2 Key Sentence Extraction

Extracting key sentences is not only an important step toward summarization, but is also useful for indexing speech archives. This is similar to the common human practice of underlining important portions of a text.

The basic scheme is to define a function to measure the importance of the sentences, and train a classifier based on the function. The machine learning approach is effective for many classification tasks in speech and language processing. However, annotation of importance is subjective and often inconsistent, and thus it is sometimes difficult to design the training scheme.

In text-based summarization studies using newspapers [32.100, 104], it is generally known that position information in articles or paragraphs is very useful because key information tends to be placed in the initial portion of the articles or paragraphs. Speech documents differ greatly from text in that they do not contain explicit structural information. Position can be measured only from the start or from the end, and this appears to be too simplistic for long speech materials such as lectures, although it is still useful for summarizing academic presentations at low summarization ratios such as 10% as described later. *Christensen* et al. [32.105] reported that the position information, which was the most effective for key sentence extraction from newspaper articles, is not nearly so effective for extraction from broadcast news.

Table 32.9 Results of key sentence extraction in the CSJ (50% summarization ratio)

Transcript	Segment	Recall	Prec.	F-measure
Manual	Manual	72.4	53.5	61.5
Manual	Auto.	72.7	46.5	56.7
Auto.	Auto.	76.1	45.5	56.9
(cf) by human subject		81.5	60.1	69.2

Kawahara et al. [32.98] proposed the use of discourse markers which suggest discourse boundaries, as described in Sect. 32.5.3, for key sentence indexing. The statistics of the discourse markers are used to determine the importance of sentences, which favors potentially section-initial sentences. This measure is also combined with the conventional tf-idf measure based on content words. Experimental results obtained using the ASR test set of the CSJ are summarized in Table 32.9. In this experiment, 37.5% of the sentences that were agreed upon by two subjects based on the 50% extraction condition were determined to be correct summaries. For reference, agreement by a third subject with this correct set was computed to estimate human performance (last row of Table 32.9). The results show that the performance of the system is only 10% lower than the human judgements. In Table 32.9, the effects of automatic transcription (ASR) and automatic sentence unit detection are also presented. It is observed that the sentence segmentation error of 20% (see Table 32.6) significantly degrades the performance, while the ASR error of 30% has little effect.

Latent semantic analysis (LSA) based on singular value decomposition (SVD) is also useful for extracting important sentences [32.106]. Each singular vector represents a salient topic, and the singular vector with the largest corresponding singular value represents the topic that is the most salient in the speech document. A fixed number of singular vectors having relatively large singular values are selected, and then for each singular vector, the sentence having the largest score is extracted as an important sentence. In this manner, the extracted sentences best describe the topics represented by the singular vectors and are semantically different from each other. The method based on the simple one-to-one mapping, however, may not generate a summary describing key information sufficiently. An enhanced method based on SVD [32.102, 107] defines an importance score in the reduced-dimensional space, and then selects sentences based on the score. Thus, the extracted sentences not only describe the significant topics but also have a latent relationship to each other.

Sameer et al. [32.108] investigated the usefulness of lexical, prosodic, structural, and discourse features in selecting extractive summaries from broadcast news stories. The lexical features include counts of named entities. The acoustic/prosodic features include speaking rate, F_0 features, log-energy features, and sentence duration. They are normalized for each speaker. The structural features include normalized sentence position and speaker type (reporter or not). The discourse features include the number of new nouns in each sentence. Experimental results showed that combination of acoustic/prosodic and structural features were the most effective.

32.6.3 Summary Generation

In an early study, summarization of dialogues within limited domains was attempted as part of the VERBMOBIL project [32.109]. *Zechner* and *Waibel* investigated how the accuracy of the summaries changes when methods for WER reduction were applied in summarizing conversations from television shows [32.110]. Recent studies on speech summarization in unrestricted domains have focused almost exclusively on broadcast news [32.101,111]. *Koumpis* and *Renals* investigated the transcription and summarization of voice mails [32.112]. Most of the previous studies on speech summarization used relatively long units, such as sentences or speaker turns, as minimal units for summarization.

Hori et al. [32.113] proposed a sentence-compaction-based summarization method, in which a set of words maximizing a summarization score is extracted from the transcript, according to a target summarization ratio. The extracted set of words is then connected to build a summary. The summarization score consists of:

- a word significance measure based on the tf-idf criterion
- a confidence measure output by ASR
- a linguistic likelihood given by n-gram probability
- a word concatenation probability determined by the dependency structure based on a stochastic dependency context-free grammar (SD-CFG)

The method was further extended to summarize a sequence of utterances, which results in a process of combining sentence extraction and compaction.

Kikuchi et al. [32.103, 114] proposed a two-stage summarization method consisting of important sentence extraction and word-based sentence compaction, as shown in Fig. 32.10. After removing fillers from the ASR output, a set of important sentences is extracted, and sentence compaction is applied to them. In the course of these processes, sentence and word units are extracted from the transcript and concatenated to produce a summary to maximize the summarization score mentioned above. The method was applied to broadcast news as well as oral presentations in the CSJ. It performed effectively for both English and Japanese speech summarization. It was also shown that sentence extraction plays a more important role than sentence compaction in improving summarization performance, especially when the summarization ratio is relatively low, say 10%.

In this context, *Hirohata* et al. [32.107] incorporated a more powerful sentence extraction method based on SVD. Sentence position information was combined to extract important sentences from the introduction and conclusion segments of each presentation. They also investigated the combination of a confidence measure and linguistic likelihood to effectively extract sentences with fewer ASR errors. This method improved the performance in the case of the 10% summarization ratio.

Because it is impossible to conduct human evaluation of automatic summarization results every time methods and parameters are changed, it is indispensable to develop objective evaluation metrics, much as is the case in machine translation research. It is still ideal to use manual summaries as the targets of the summarization system. Since the manual summaries vary according to human subjects, the way in which these variations are handled is an important problem.

Hori et al. [32.115] proposed to merge all of the human summaries into a single word network, which is considered to cover approximately all possible correct summaries. The word accuracy of the automatic

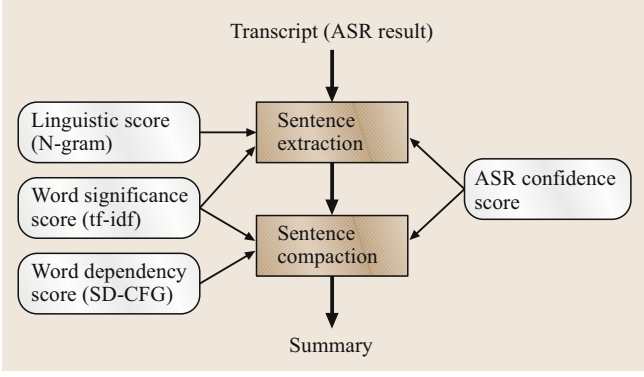


Fig. 32.10 Flow of two-stage speech summarization

summary is then measured as the summarization accuracy, SumACCY, by comparing the word sequence with the closest sequence extracted from the word network. This metric works reasonably well at relatively high summarization ratios such as 50%, but has problems at low summarization ratios such as 10%, because the variation between manual summaries is so large that the network accepts inappropriate summaries. Therefore, they have also proposed the use of the word accuracy obtained by using the manual summaries individually, SumACCY-E. In this metric, the largest score of SumACCY-E among all human summaries (SumACCY-E/max) or its average score (SumACCY-E/ave) is used.

ROUGE, which is the most widely used metric, is the *n*-gram recall between an automatic summary and a set of manual summaries [32.116]. The number of co-occurrences of 1-, 2-, or 3-grams in the manual summary and the automatic summary is usually used for this computation.

Hirohata et al. [32.107] investigated and evaluated these objective evaluation metrics in the framework of sentence extraction-based speech summarization for academic presentations with the condition of 10% summarization ratio. Correlation analysis between subjective and objective evaluation scores confirmed that the summarization accuracy and the ROUGE based on 2- or 3-gram recall were effective evaluation metrics.

32.7 Conclusions

We have reviewed the major research efforts concerning spontaneous speech processing, including speech recognition, metadata extraction, and speech summarization. Although many of the results were obtained using the corpus of spontaneous Japanese (CSJ), most of the conclusions should apply to all languages.

Spontaneous speech recognition requires a different paradigm from that of conventional ASR, in that a faithful transcript is not necessarily useful because of the existence of disfluency phenomena and the lack of punctuation and line breaking. Although it is arguable whether or not we should simply remove disfluencies, intelligent transcription, as performed by human transcribers or stenographers, must include the deletion of fillers and cleaning of self-repairs, which are addressed in Sect. 32.5. Moreover, it should also involve the recovery of omitted particles and the correction of colloquial expressions. There is a potentially huge demand for these types of intelligent transcription technologies, but the current level of ASR accuracy, which is approximately 70–80%, is not satisfactory. Thus, more studies on modeling and adaptation of acoustic and language models are needed, by considering the complex factors that affect the ASR performance in spontaneous speech. We should also explore a scheme that integrates the cleaning postprocess, or directly minimizes the edit distance with the cleaned transcript. An approach based on weighted finite-state transducers (WFST) is interesting for this purpose [32.117].

Spontaneous speech processing will expand the application of ASR technologies. First, it will make possible automatic indexing and metadata annotation for content-based access to huge audio archives, enabling text-based information retrieval and extraction techniques to be applied to audio materials. Next, it is also useful as a means of filtering speech documents for effective presentation in various ways, for example, by changing the summarization ratio for a given speech document. It has been shown that current technology can be useful for information retrieval or key-sentence extraction, and that ASR errors of 20–30% do not affect performance greatly. For further improvement, an integrated scheme should also be investigated. Minimum Bayes-risk (MBR) decoding can be formulated by considering the importance of the words in information retrieval or summarization [32.118].

In the future, integration with other media, such as video, should also be studied more intensively. Furthermore, integration with knowledge processing must be explored, because speech is a media for exchanging knowledge and is itself thus a source of knowledge.

References

32.1 D.S. Pallet, J.G. Fiscus, J.S. Garofolo, A. Martin, M.A. Przybocki: Broadcast news benchmark test results, In DARPA Broadcast News Workshop (1999) pp. 5–12

32.2 B. Chen, L.-S. Lee: Spoken document understanding and organization, Signal Process. Mag. **22**(5), 42–60 (2005)

32.3 C. Cieri, D. Graff, M. Liberman, N. Martey, S. Strassel: The TDT-2 text and speech corpus, DARPA Broadcast News Workshop (1999) pp. 57–60

32.4 E. Leeuwis, M. Federico, M. Cettolo: Language modeling and transcription of the TED corpus lectures, Proc. IEEE-ICASSP, Vol. 1 (2003) pp. 232–235

32.5 S. Furui: Recent advances in spontaneous speech recognition and understanding, Proc. ISCA and IEEE Workshop on Spontaneous Speech Processing and Recognition, ed. by (2003) pp. 1–4

32.6 A. Park, T. Hazen, J. Glass: Automatic processing of audio lectures for information retrieval: Vocabulary selection and language modeling, In Proc. IEEE-ICASSP, Vol. 1 (2005) pp. 497–500

32.7 B. Ramabhadran, J. Huang, M. Picheny: Towards automatic transcription of large spoken archives – English ASR for the MALACH project, Proc. IEEE-ICASSP, Vol. 1 (2003) pp. 216–219

32.8 J. Fiscus, W.M. Fisher, A. Martin, M. Przybocki, D.S. Pallett: 2000 NIST evaluation of conversational speech recognition over the telephone, DARPA Speech Transcription Workshop (2000)

32.9 S.F. Chen, B. Kingsbury, L. Mangu, D. Povey, G. Saon, H. Soltau, G. Zweig: Advances in speech transcription at IBM under the DARPA EARS program, IEEE Trans. Audio, Speech Lang. Process **14**(5), 1596–1608 (2006)

32.10 S. Matsoukas, J.-L. Gauvain, G. Adda, T. Colthurst, C.-L. Kao, O. Kimball, L. Lamel, F. Lefevre, J.Z. Ma, J. Makhoul, L. Nguyen, R. Prasad, R. Schwartz, H. Schwenk, B. Xiang: Advances in transcription of broadcast news and conversational telephone speech within the combined EARS BBN/LIMSI system, IEEE Trans. Audio, Speech Lang. Process **14**(5), 1541–1556 (2006)

32.11 J.J. Godfrey, E.C. Holliman, J. McDaniel: SWITCHBOARD: Telephone speech corpus for research and development, Proc. IEEE-ICASSP, Vol. 1 (1992) pp. 517–520

32.12 H. Soltau, B. Kingsbury, L. Mangu, D. Povey, G. Saon, G. Zweig: The IBM 2004 conversational telephony system for rich transcription, Proc. IEEE-ICASSP, Vol. 1 (2005) pp. 205–208

32.13 J.S. Garofol, C.D. Laprun, J.G. Fiscus: The RT-04 spring meeting recognition evaluation, NIST Meeting Recognition Workshop (2004)

32.14 J. Fiscus, J. Ajot, J.S. Garofol: The rich transcription 2006 evaluation overview and speech-to-text results, NIST Meeting Recognition Workshop (2006)

32.15 C. Gollan, M. Bisani, S. Kanthak, R. Schluter, H. Ney: Cross domain automatic transcription on the TC-STAR EPPS corpus, Proc. IEEE-ICASSP, Vol. 1 (2005) pp. 825–828

32.16 J. Loeoef, M. Bisani, C. Gollan, G. Heigold, B. Hoffmeister, C. Plahl, R. Schlueter, H. Ney: The 2006 RWTH parliamentary speeches transcription system, Proc. TC-STAR Workshop on Speech-to-Speech Translation (2006) pp. 133–138

32.17 Y. Akita, C. Troncoso, T. Kawahara: Automatic transcription of meetings using topic-oriented language model adaptation, Proc. Western Pacific Acoustics Conference (WESPAC) (2006)

32.18 D. McKelvie, A. Isard, A. Mengel, M.B. Moller, M. Grosse, M. Klein: The MATE workbench – an annotation tool for XML coded speech corpora, Speech Commun. **33**, 97–112 (2001)

32.19 S. Bird, M. Liberman: A formal framework for linguistic annotation, Speech Commun. **33**, 23–60 (2001)

32.20 L. Lamel, J. Gauvain, G. Adda: Investigating lightly supervised acoustic model training, Proc. IEEE-ICASSP, Vol. 1 (2001) pp. 477–480

32.21 L. Nguyen, B. Xiang: Light supervision in acoustic model training, Proc. IEEE-ICASSP, Vol. 1 (2004) pp. 185–188

32.22 K. Maekawa: Corpus of Spontaneous Japanese: Its design and evaluation, Proc. ISCA and IEEE Workshop on Spontaneous Speech Processing and Recognition (2003) pp. 7–12

32.23 K. Uchimoto, C. Nobata, A. Yamada, S. Sekine, H. Isahara: Morphological analysis of corpus of spontaneous Japanese, Proc. ISCA and IEEE Workshop on Spontaneous Speech Processing and Recognition (2003) pp. 159–162

32.24 K. Maekawa, H. Kikuchi, Y. Igarashi, J. Venditti: X-JToBI: an Extended J-ToBI for Spontaneous Speech, Proc. ICSLP (2002) pp. 1545–1548

32.25 S. Furui, M. Nakamura, T. Ichiba, K. Iwano: Analysis and recognition of spontaneous speech using Corpus of Spontaneous Japanese, Speech Commun. **47**, 208–219 (2005)

32.26 E. Fosler, M. Weintraub, S. Wegmann, Y.H. Kao, S. Khudanpur, C. Galles, M. Saraclar: Automatic learning of word pronunciation from data, Proc. ICSLP (1996)

32.27 T. Shinozaki, S. Furui: Analysis on individual differences in automatic transcription of spontaneous presentations, Proc. IEEE-ICASSP, Vol. 1 (2002) pp. 729–732

32.28 L. Lee, R.C. Rose: Speaker normalization using efficient frequency warping procedures, Proc. IEEE-ICASSP (1996) pp. 353–356

32.29 S. Wegmann, D. McAllaster, J. Orloff, B. Peskin: Speaker normalization on conversational telephone speech, Proc. IEEE-ICASSP (1996) pp. 339–342

32.30 J.W. McDonough, T. Anastasakos, G. Zavaliagkos, H. Gish: Speaker-adapted training on the switchboard corpus, Proc. IEEE-ICASSP (1997) pp. 1059–1062

32.31 D. Pye, P.C. Woodland: Experiments in speaker normalisation and adaptation for large vocabulary speech recognition, Proc. IEEE-ICASSP (1997) pp. 1047–1050

32.32 M. Ostendorf, V.V. Digalakis, O.A. Kimball: From HMM's to segment models: A unified view of stochastic modeling for speech recognition,

IEEE Trans. Speech Audio Process. **4**(5), 360–378 (1996)
32.33 Y. Gong, J.-P. Haton: Stochastic trajectory modeling for speech recognition, Proc. IEEE-ICASSP, Vol. 1 (1994) pp. 57–60
32.34 C. Fugen, I. Rogina: Integrating dynamic speech modalities into context decision trees, Proc. IEEE-ICASSP (2000) pp. 1277–1280
32.35 G. Zweig, S. Russell: Probabilistic modeling with Bayesian networks for automatic speech recognition, Proc. ICSLP (1998) pp. 3011–3014
32.36 J.A. Bilmes: Buried Markov models for speech recognition, Proc. ICASSP, Vol. 2 (1999) pp. 713–716
32.37 J. Bilmes, C. Bartels: Graphical model architectures for speech recognition, Signal Process. Mag. **22**(5), 89–100 (2005)
32.38 J. Zheng, H. Franco, F. Weng: Word-level rate of speech modeling using rate-specific phones and pronunciations, Proc. IEEE-ICASSP (2000) pp. 1775–1778
32.39 N. Morgan, E. Fosler, N. Mirghafori: Speech recognition using on-line estimation of speaking rate, Proc. EUROSPEECH (1997) pp. 2079–2082
32.40 H. Nanjo, K. Kato, T. Kawahara: Speaking rate dependent acoustic modeling for spontaneous lecture speech recognition, Proc. EUROSPEECH (2001) pp. 2531–2534
32.41 J. Nedel, R. Stern: Duration normalization for improved recognition of spontaneous and read speech via missing feature methods, Proc. IEEE-ICASSP, Vol. 1 (2001) pp. 313–316
32.42 M. Richardson, M. Hwang, A. Acero, X.D. Huang: Improvements on speech recognition for fast talkers, Proc. EUROSPEECH (1999) pp. 411–414
32.43 S. Tsuge, T. Fukuda, K. Kita: Frame-period adaptation for speaking rate robust speech recognition, Proc. ICSLP, Vol. 3 (2000) pp. 718–721
32.44 K. Okuda, T. Kawahara, S. Nakamura: Speaking rate compensation based on likelihood criterion in acoustic model training and decoding, Proc. ICSLP (2002) pp. 2589–2592
32.45 T. Shinozaki, S. Furui: Hidden mode HMM using Bayesian network for modeling speaking rate fluctuation, Proc. IEEE Workshop Automatic Speech Recognition and Understanding (2003)
32.46 H. Nanjo, T. Kawahara: Language model and speaking rate adaptation for spontaneous presentation speech recognition, IEEE Trans. Speech and Audio Process. **12**(4), 391–400 (2004)
32.47 T. Holter, T. Svendsen: Combined optimisation of baseforms and model parameters in speech recognition based on acoustic subword units, Proc. IEEE Workshop Automatic Speech Recognition and Understanding (1997) pp. 199–206
32.48 M. Bacchiani, M. Ostendorf: Using automatically-derived acoustic sub-word units in large vocabulary speech recognition, Proc. ICSLP (1998) pp. 1843–1846
32.49 T. Sloboda, A. Waibel: Dictionary learning for spontaneous speech recognition, Proc. ICSLP, Vol. 4 (1996) pp. 2328–2331
32.50 M. Finke, A. Waibel: Speaking mode dependent pronunciation modeling in large vocabulary conversational speech recognition, Proc. EUROSPEECH (1997) pp. 2379–2382
32.51 E. Fosler-Lussier: Multi-level decision trees for static and dynamic pronunciation models, Proc. EUROSPEECH (1999) pp. 463–466
32.52 J.M. Kessens, M. Wester, H. Strik: Improving the performance of a Dutch CSR by modeling within-word and cross-word pronunciation variation, Speech Communication **29**, 193–207 (1999)
32.53 B. Peskin, M. Newman, D. McAllaster, V. Nagesha, H.B. Richards, S. Wegmann, M. Hunt, L. Gillick: Improvements in recognition of conversational telephone speech, Proc. IEEE-ICASSP (1999) pp. 53–56
32.54 H. Schramm, X. Aubert: Efficient integration of multiple pronunciations in a large vocabulary decoder, Proc. IEEE-ICASSP (2000) pp. 1659–1662
32.55 M. Riley, W. Byrne, M. Finke, S. Khudanpur, A. Ljolje, J. McDonough, H. Nock, M. Saraclar, C. Wooters, G. Zavaliagkos: Stochastic pronunciation modelling from hand-labelled phonetic corpora, Speech Commun. **29**, 209–224 (1999)
32.56 T. Fukada, T. Yoshimura, Y. Sagisaka: Automatic generation of multiple pronunciations based on neural networks, Speech Commun. **27**, 63–73 (1999)
32.57 D. Torre, L. Villarrubia, J.M. Elvira, L. Hernandez-Gomez: Automatic alternative transcription generation and vocabulary selection for flexible word recognizers, Proc. IEEE-ICASSP (1997) pp. 1463–1466
32.58 Y. Akita, T. Kawahara: Generalized statistical modeling of pronunciation variations using variable-length phone context, Proc. IEEE-ICASSP, Vol. 1 (2005) pp. 689–692
32.59 X. Zhu, R. Rosenfeld: Improving trigram language modeling with the world wide web, Proc. IEEE-ICASSP, Vol. 1 (2001) pp. 533–536
32.60 R. Sarikaya, A. Gravano, Y. Gao: Rapid language model development using external resources for new spoken dialog domains, Proc. IEEE-ICASSP, Vol. 1 (2005) pp. 573–576
32.61 T. Ng, M. Ostendorf, M.-Y. Hwang, M. Siu, I. Bulyko, X. Lei: Web-data augmented language models for Mandarin conversational speech recognition, Proc. IEEE-ICASSP, Vol. 1 (2005) pp. 589–592
32.62 T. Misu, T. Kawahara: A bootstrapping approach for developing language model of new spoken dialogue systems by selecting web texts, In Proc. INTERSPEECH (2006) pp. 9-12
32.63 Y. Akita, T. Kawahara: Efficient estimation of language model statistics of spontaneous speech via statistical transformation model, Proc. IEEE-ICASSP, Vol. 1 (2006) pp. 1049–1052

32.64 R. Khun, R. De Mori: A cache-based natural language model for speech recognition, IEEE Trans. Pattern Analysis and Machine Intelligence **12**(6), 570–583 (1990)

32.65 P. Clarkson, A.J. Robinson: Language model adaptation using mixtures and an exponentially decaying cache, Proc. IEEE-ICASSP (1997) pp. 799–802

32.66 R. Lau, R. Rosenfeld, S. Roukos: Trigger-based language models: A maximum entropy approach, Proc. IEEE-ICASSP, Vol. 2 (1993) pp. 45–48

32.67 C. Troncoso, T. Kawahara: Trigger-based language model adaptation for automatic meeting transcription, Proc. INTERSPEECH (2005) pp. 1297–1300

32.68 T. Yokoyama, T. Shinozaki, K. Iwano, S. Furui: Unsupervised class-based language model adaptation for spontaneous speech recognition, Proc. IEEE-ICASSP, Vol. 1 (2003) pp. 236–239

32.69 G. Moore, S. Young: Class-based language model adaptation using mixtures of weights, Proc. ICSLP, Vol. 4 (2000) pp. 512–515

32.70 L. Lussier, E.W.D. Whittaker, S. Furui: Unsupervised language model adaptation methods for spontaneous speech, Proc. ICSLP (2004) pp. 1981–1984

32.71 L. Chen, J.L. Gauvain, L. Lamel, G. Adda, M. Adda: Using information retrieval methods for language model adaptation, Proc. EUROSPEECH (2001) pp. 255–258

32.72 R. Kneser, V. Steinbiss: On the dynamic adaptation of stochastic language models, Proc. IEEE-ICASSP, Vol. 2 (1993) pp. 586–589

32.73 J.R. Bellegarda: A multispan language modeling framework for large vocabulary speech recognition, IEEE Trans. Speech and Audio Process. **6**(5), 468–475 (1998)

32.74 T. Hoffman: Probabilistic latent semantic indexing, Proc. SIG-IR (1999)

32.75 Y. Akita, T. Kawahara: Language model adaptation based on PLSA of topics and speakers, Proc. ICSLP (2004) pp. 1045–1048

32.76 A. Stolcke, E. Shriberg, R. Bates, M. Ostendorf, D. Hakkani, M. Plauche, G. Tur, Y. Lu: Automatic detection of sentence boundaries and disfluencies based on recognized words, Proc. ICSLP (1998) pp. 2247–2250

32.77 Y. Liu, E. Shriberg, A. Stolcke, B. Peskin, J. Ang, D. Hillard, M. Ostendorf, M. Tomalin, P. Woodland, M. Harper: Structural metadata research in the EARS program, Proc. IEEE-ICASSP, Vol. 5 (2005) pp. 957–960

32.78 E. Shriberg: Spontaneous speech: How people really talk and why engineers should care, Proc. INTERSPEECH (2005) pp. 1781–1784

32.79 T. Kudo, Y. Matsumoto: Chunking with support vector machines, Proc. NAACL (2001)

32.80 A.L. Berger, S.A. Della Pietra, V.J. Della Pietra: A maximum entropy approach to natural language processing, Comput. Ling. **22**(1), 39–71 (1996)

32.81 J. Huang, G. Zweig: Maximum entropy model for punctuation annotation from speech, Proc. ICSLP (2002) pp. 917–920

32.82 J. Lafferty, A. McCallum, F. Pereira: Conditional random fields: Probabilistic models for segmenting and labeling sequence data, Proc. Int. Conf. Machine Learning (2001) pp. 282–289

32.83 E. Shriberg, A. Stolcke, D. Hakkani-Tur, G. Tur: Prosody-based automatic segmentation of speech into sentences and topics, Speech Commun. **32**, 127–154 (2000)

32.84 Y. Akita, M. Saikou, H. Nanjo, T. Kawahara: Sentence boundary detection of spontaneous Japanese using statistical language model and support vector machines, Proc. INTERSPEECH (2006) pp. 1033–1036

32.85 Y. Liu, E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf, M. Harper: Enriching speech recognition with automatic detection of sentence boundaries and disfluencies, IEEE Trans. Audio, Speech Lang. Process. **14**(5), 1526–1540 (2006)

32.86 Y. Liu, A. Stolcke, E. Shriberg, M. Harper: Comparing and combining generative and posterior probability models: Some advances in sentence boundary detection in speech, Proc. EMNLP (2004) pp. 64–71

32.87 M. Tomalin, P.C. Woodland: Discriminatively trained Gaussian mixture models for sentence boundary detection, Proc. IEEE-ICASSP, Vol. 1 (2006) pp. 549–552

32.88 J.H. Kim, P.C. Woodland: The use of prosody in a combined system for punctuation generation and speech recognition, Proc. EUROSPEECH (2001) pp. 2757–2760

32.89 K. Shitaoka, K. Uchimoto, T. Kawahara, H. Isahara: Dependency structure analysis and sentence boundary detection in spontaneous Japanese, Proc. COLING (2004) pp. 1107–1113

32.90 R. Hamabe, K. Uchimoto, T. Kawahara, H. Isahara: Detection of quotations and inserted clauses and its application to dependency structure analysis in spontaneous Japanese, Proc. COLING-ACL volume Poster Sessions (2006) pp. 324–330

32.91 F.M. Quimbo, T. Kawahara, S. Doshita: Prosodic analysis of fillers and self-repair in Japanese speech, Proc. ICSLP (1998) pp. 3313–3316

32.92 C. Nakatani, J. Hirschberg: A speech first model for repair detectin and correction, Proc. ARPA Human Language Technology Workshop (1993) pp. 329–334

32.93 Y. Liu, E. Shriberg, A. Stolcke, M. Harper: Comparing HMM maximum entropy and conditional random fields for disfluency detection, Proc. INTERSPEECH (2005) pp. 3033–3036

32.94 P.A. Heeman, J.F. Allen: Speech repairs, intonational phrases and discourse markers: Modeling speakers' utterances in spoken dialog, Comput. Ling. **25**(4), 527–571 (1999)

32.95 M. Honal, T. Schultz: Correction of disfluencies in spontaneous speech using asnoisy-channel approach, Proc. EUROSPEECH (2003) pp. 2781–2784

32.96 R.J. Passonneau, D.J. Litman: Discourse segmentation by human and automated means, Comput. Ling. **23**(1), 103–139 (1997)

32.97 M. Haase, W. Kriechbaum, G. Mohler, G. Stenzel: Deriving document structure from prosodic cues, Proc. EUROSPEECH (2001) pp. 2157–2160

32.98 T. Kawahara, M. Hasegawa, K. Shitaoka, T. Kitade, H. Nanjo: Automatic indexing of lecture presentations using unsupervised learning of presumed discourse markers, IEEE Trans. Speech Audio Process. **12**(4), 409–419 (2004)

32.99 K. Zechner: Spoken language condensation in the 21st century, Proc. EUROSPEECH (2003) pp. 1989–1992

32.100 I. Mani, M. Maybury: *Advances in Automatic Text Summarization* (MIT Press, Cambridge 1999)

32.101 R. Valenza, T. Robinson, M. Hickey, R. Tucker: Comparing and combining generative and posterior probability models: Some advances in sentence boundary detection in speech, ESCA Workshop Accessing Information in Spoken Audio (1999) pp. 111–116

32.102 G. Murray, S. Renals, J. Carletta: Extractive summarization of meeting recordings, Proc. EUROSPEECH (2005) pp. 593–596

32.103 S. Furui, T. Kikuchi, Y. Shinnaka, C. Hori: Speech-to-text and speech-to-speech summarization of spontaneous speech, IEEE Trans. Speech Audio Process. **12**(4), 401–408 (2004)

32.104 J. Kupiec, J.O. Pedersen, F. Chen: A trainable document summarizer, Proc. SIG-IR (1995)

32.105 H. Christensen, Y. Gotoh, B. Kolluru, S. Renals: Are extractive text summarization techniques portable to broadcast news?, Proc. IEEE Workshop Automatic Speech Recognition and Understanding (2003)

32.106 Y. Gong, X. Liu: Generic text summarization using relevance measure and latent semantic analysis, Proc. SIG-IR (2001)

32.107 M. Hirohata, Y. Shinnaka, K. Iwano, S. Furui: Sentence extraction-based presentation summarization techniques and evaluation metrics, Proc. IEEE-ICASSP, Vol. 1 (2005) pp. 1065–1068

32.108 M. Sameer, J. Hirschberg: Comparing lexical, acoustic/prosodic, structural and discourse features for speech summarization, Proc. INTERSPEECH (2005) pp. 621–624

32.109 J. Alexandersson, P. Poller: Towards multilingual protocol generation for spontaneous dialogues, Proc. INLG-98 (1998)

32.110 K. Zechner, A. Waibel: Minimizing word error rate in textual summaries of spoken language, Proc. NAACL (2000)

32.111 J.S. Garofolo, E.M. Voorhees, C.G.P. Auzanne, V.M. Stanford: Spoken document retrieval: 1998 evaluation and investigation of new metrics, ESCA Workshop Accessing Information in Spoken Audio (1999) pp. 1–7

32.112 K. Koumpis, S. Renals: Transcription and summarization of voicemail speech, Proc. ICSLP, Vol. 2 (2000) pp. 688–691

32.113 C. Hori, S. Furui: Advances in speech summarization, Proc. EUROSPEECH (2001) pp. 1771–1774

32.114 T. Kikuchi, S. Furui, C. Hori: Automatic speech summarization based on sentence extraction and compaction, Proc. IEEE-ICASSP, Vol. 1 (2003) pp. 384–387

32.115 C. Hori, T. Hirao, H. Isozaki: Evaluation measures considering sentence concatenation for automatic summarization by sentence or word extraction, Proc. ACL Workshop Text Summarization Branches Out (2004) pp. 82–88

32.116 C.-Y. Lin: ROUGE: A package for automatic evaluation of summaries, Proc. ACL Workshop Text Summarization Branches Out (2004) pp. 74–81

32.117 T. Hori, C. Hori, Y. Minami: Speech summarization using weighted finite-state transducers, Proc. EUROSPEECH (2003) pp. 2817–2820

32.118 H. Nanjo, T. Kawahara: A new ASR evaluation measure and minimum Bayes-risk decoding for open-domain speech understanding, Proc. IEEE-ICASSP, Vol. 1 (2005) pp. 1053–1056

36. Overview of Speaker Recognition

A. E. Rosenberg, F. Bimbot, S. Parthasarathy

An introduction to automatic speaker recognition is presented in this chapter. The identifying characteristics of a person's voice that make it possible to automatically identify a speaker are discussed. Subtasks such as speaker identification, verification, and detection are described. An overview of the techniques used to build speaker models as well as issues related to system performance are presented. Finally, a few selected applications of speaker recognition are introduced to demonstrate the wide range of applications of speaker recognition technologies. Details of text-dependent and text-independent speaker recognition and their applications are covered in the following two chapters.

36.1 Speaker Recognition

36.1.1 Personal Identity Characteristics

Human beings have many characteristics that make it possible to distinguish one individual from another. Some individuating characteristics can be perceived very readily such as facial features and vocal qualities and behavior. Others, such as fingerprints, iris patterns, and DNA structure are not readily perceived and require measurements, often quite complex measurements, to capture distinguishing characteristics. In recent years biometrics has emerged as an applied scientific discipline with the objective of automatically capturing personal identifying characteristics and using the measurements for security, surveillance, and forensic applications [36.1]. Typical applications using biometrics secure transactions, information, and premises to authorized individuals. In surveillance applications, the goal is to detect and track a target individual among a set of nontarget individuals. In forensic applications a sample of biometric measurements is obtained from an unknown individual, the perpetrator. The task is to compare this sample with a database of similar measurements from known individuals to find a match.

Many personal identifying characteristics are based on physiological properties, others on behavior, and some combine physiological and behavioral properties. From the point of view of using personal identity characteristics as a biometric for security, physiological characteristics may offer more intrinsic security since they are not subject to the kinds of voluntary variations found in behavioral features. Voice is an example of a biometric that combines physiological and behav-

ioral characteristics. Voice is attractive as a biometric for many reasons. It can be captured non-intrusively and conveniently with simple transducers and recording devices. It is particularly useful for remote-access transactions over telecommunication networks. A drawback is that voice is subject to many sources of variability, including behavioral variability, both voluntary and involuntary. An example of involuntary variability is a speaker's inability to repeat utterances precisely the same way. Another example is the spectral changes that occur when speakers vary their vocal effort as background noise increases. Voluntary variability is an issue when speakers attempt to disguise their voices. Other sources of variability include physical voice variations due to respiratory infections and congestion. External sources of variability are especially problematic, including variations in background noise, and transmission and recording characteristics.

36.1.2 Speaker Recognition Definitions

Different tasks are defined under the general heading of speaker recognition. They differ mainly with respect to the kind of decision that is required for each task. In speaker identification a voice sample from an unknown speaker is compared with a set of labeled speaker models. When it is known that the set of speaker models includes all speakers of interest the task is referred to as *closed-set* identification. The label of the best matching speaker is taken to be the identified speaker. Most speaker identification applications are *open-set*, meaning that it is possible that the unknown speaker is not included in the set of speaker models. In this case, if no satisfactory match is obtained, a *no-match* decision is provided.

In a speaker verification trial an identity claim is provided or asserted along with the voice sample. In this case, the unknown voice sample is compared only with the speaker model whose label corresponds to the identity claim. If the quality of the comparison is satisfactory, the identity claim is accepted; otherwise the claim is rejected. Speaker verification is a special case of open-set speaker identification with a one-speaker target set. The speaker verification decision mode is intrinsic to most access control applications. In these applications, it is assumed that the claimant will respond to prompts cooperatively.

It can readily be seen that in the speaker identification task performance degrades as the number of speaker models and the number of comparisons increases. In a speaker verification trial only one comparison is required, so speaker verification performance is independent of the size of the speaker population.

A third speaker recognition task has been defined in recent years in National Institute of Standards and Technology (NIST) speaker recognition evaluations; it is generally referred to as speaker detection [36.2, 3]. The NIST task is an open-set identification decision associated exclusively with conversational speech. In this task an unknown voice sample is provided and the task is to determine whether or not one of a specified set of known speakers is present in the sample. A complicating factor for this task is that the unknown sample may contain speech from more than one speaker, such as in the summed two sides of a telephone conversation. In this case, an additional task called speaker tracking is defined, in which it is required to determine the intervals in the test sample during which the detected speaker is talking. In other applications where the speech samples are multispeaker, speaker tracking has also been referred to as speaker segmentation, speaker indexing, and speaker diarization [36.4–10]. It is possible to cast the speaker segmentation task as an acoustical change detection task without creating models. The time instants where a significant acoustic change occurs are assumed to be the boundaries between different speaker segments. In this case, in the absence of speaker models, speaker segmentation would not be considered a speaker recognition task. However, in most reported approaches to this task some sort of speaker modeling does take place. The task usually includes labeling the speaker segments. In this case the task falls unambiguously under the speaker recognition heading.

In addition to decision modes, speaker recognition tasks can be categorized by the kind of speech that is input. If the speaker is prompted or expected to provide a known text and if speaker models have been trained explicitly for this text, the input mode is said to be text dependent. If, on the other hand, the speaker cannot be expected to utter specified texts the input mode is text independent. In this case speaker models are not trained on explicit texts.

36.1.3 Bases for Speaker Recognition

The principal function associated with the transmission of a speech signal is to convey a message. However, along with the message, additional kinds of information are transmitted. These include information about the gender, identity, emotional state, health, etc. of the speaker. The source of all these kinds of information lie in both physiological and behavioral characteristics.

The physiological features are shown in Fig. 36.1 showing a cross-section of the human vocal tract. The shape of the vocal tract, determined by the position of articulators, the tongue, jaw, lips, teeth, and velum, creates a set of acoustic resonances in response to periodic puffs of air generated by the glottis for voiced sounds or aperiodic excitation caused by air passing through tight constrictions in the vocal tract. The spectral peaks associated with periodic resonances are referred to as speech formants. The locations in frequency and, to a lesser degree, the shapes of the resonances distinguish one speech sound from another. In addition, formant locations and bandwidths and spectral differences associated with the overall size of the vocal tract serve to distinguish the same sounds spoken by different speakers. The shape of the nasal tract, which determines the quality of nasal sounds, also varies significantly from speaker to speaker. The mass of the glottis is associated with the basic fundamental frequency for voiced speech sounds. The average basic fundamental frequency is approximately 100 Hz for adult males, 200 Hz for adult females, and 300 Hz for children. It also varies from individual to individual.

Speech signal events can be classified as segmental or suprasegmental. Generally, segmental refers to the features of individual sounds or segments, whereas suprasegmental refers to properties that extend over several speech sounds. Speaking behavior is associated with the individual's control of articulators for individual speech sounds or segments and also with suprasegmental characteristics governing how individual speech sounds are strung together to form words. Higher-level speaking behavior is associated with choices of words and syntactic units. Variations in fundamental frequency or pitch and rhythm are also higher-level features of the speech signal along with such qualities as breathiness, strength of vocal effort, etc. All of these vary significantly from speaker to speaker.

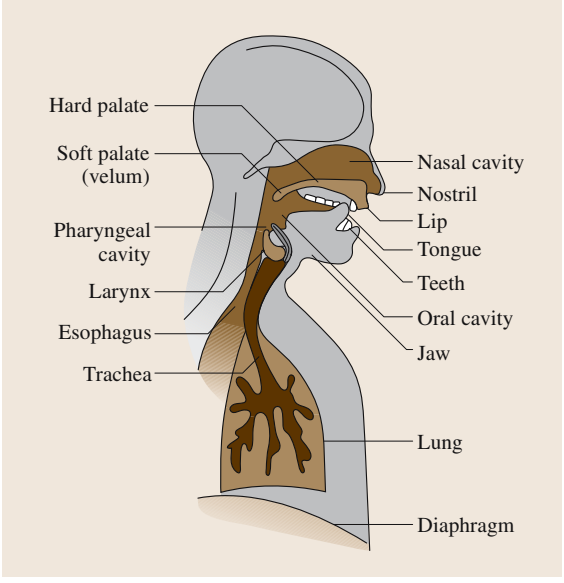


Fig. 36.1 Physiology of the human vocal tract (Reproduced with permission from L. H. *Jamieson* [36.11])

36.1.4 Extracting Speaker Characteristics from the Speech Signal

A perceptual view classifies speech as containing *low-level* and *high-level* kinds of information. Low-level features of speech are associated with the periphery in the brain's perception of speech and are relatively accessible from the speech signal. High-level features are associated with more-central locations in the perception mechanism. Generally speaking, low-level speaker features are easier to extract from the speech signal and model than high-level features. Many such features are associated with spectral correlates such as formant locations and bandwidths, pitch periodicity, and segmental timings. High-level features include the perception of words and their meaning, syntax, prosody, dialect, and idiolect.

It is not easy to extract stable and reliable formant features explicitly from the speech signal. In most instances it is easier to carry out short-term spectral amplitude measurements that capture low-level speaker characteristics implicitly. Short-term spectral measurements are typically carried out over 20–30 ms windows and advanced every 10 ms. Short speech sounds have durations less than 100 ms whereas stressed vowel sounds can last for 300 ms or more. Advancing the time window every 10 ms enables the temporal characteristics of individual speech sounds to be tracked and the 30 ms analysis window is usually sufficient to provide good spectral resolution of these sounds and at the same time short enough to resolve significant temporal characteristics. There are two principal methods of short-term spectral analysis, filter bank analysis and linear predictive coding (LPC) analysis. In filter bank analysis the speech signal is passed through a bank of bandpass filters covering a range of frequencies consistent with the transmission characteristics of the signal. The spacing of the filters can be uniform or, more likely, spaced nonuniformly, consistent with perceptual criteria such as the mel or bark scale [36.12], which provides a linear spacing in frequency below 1000 Hz

and logarithmic spacing above. The output of each filter is typically implemented as a windowed, short-term Fourier transform using fast Fourier transform (FFT) techniques. This output is subject to a nonlinearity and low-pass filter to provide an energy measurement. LPC-derived features almost always include regression measurements that capture the temporal evolution of these features from one speech segment to another. It is no accident that short-term spectral measurements are also the basis for speech recognizers. This is because an analysis that captures the differences between one speech sound and another can also capture the difference between the same speech sound uttered by different speakers, often with resolutions surpassing human perception.

Other measurements that are often carried out are correlated with prosody such as pitch and energy tracking. Pitch or periodicity measurements are relatively easy to make. However, periodicity measurement is meaningful only for voiced speech sounds so it is necessary also to have a detector that can discriminate voiced from unvoiced sounds. This complication often makes it difficult to obtain reliable pitch tracks over long-duration utterances.

Long-term average spectral and fundamental frequency measurements have been used in the past for speaker recognition, but since these measurements provide feature averages over long durations they are not capable of resolving detailed individual differences.

Although computational ease is an important consideration for selecting speaker-sensitive feature measurements, equally important considerations are the stability of the measurements, including whether they are subject to variability, noise, and distortions from one measurement of a speaker's utterances to another. One source of variability is the speaker himself. Features that are correlated with behavior such as pitch contours – pitch measured as a function of time over specified utterances – can be consciously varied from one token of an utterance to another. Conversely, cooperative speakers can control such variability. More difficult to deal with are the variability and distortion associated with recording environments, microphones, and transmission media. The most severe kinds of variability problems occur when utterances used to train models are recorded under one set of conditions and test utterances are recorded under another.

A block diagram of a speaker recognition is shown in Fig. 36.2, showing the basic elements discussed in this section. A sample of speech from an unknown speaker is input to the system. If the system is a speaker verification system, an identity claim or assertion is also input. The speech sample is recorded, digitized, and analyzed. The analysis is typically some sort of short-term spectral analysis that captures speaker-sensitive features as described earlier in this section. These features are compared with prototype features compiled into the models of known speakers. A matching process is invoked to compare the sample features and the model features. In the case of closed-set speaker identification, the match is assigned to the model with the best matching score. In the case of speaker verification, the matching score is compared with a predetermined threshold to decide whether to accept or reject the identity claim. For open-set identification, if the matching score for the best matching model does not pass a threshold test, a no-match decision is made.

36.1.5 Applications

As mentioned, the most widespread applications for automatic speaker recognition are for security. These are typically speaker verification applications intended to control access to privileged transactions or information remotely over a telecommunication network. These are usually configured in a text-dependent mode in which customers are prompted to speak personalized verification phrases such as personal identification numbers

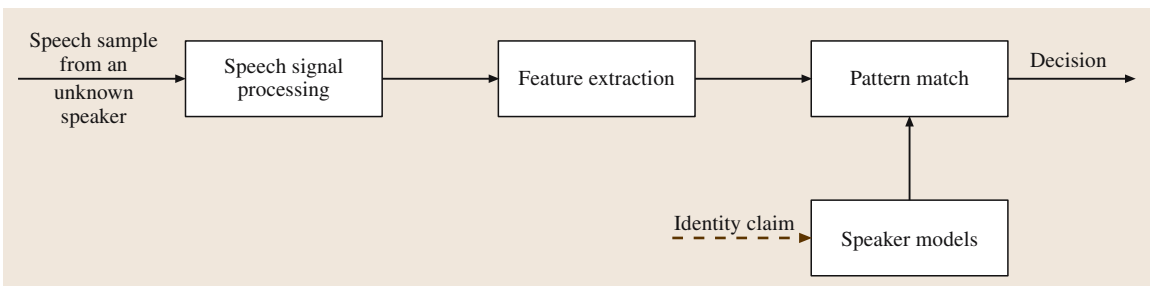


Fig. 36.2 Block diagram of a speaker recognition system

(PINs) spoken as a string of digits. Typically, PIN utterances are decoded using a speaker-independent speech recognizer to provide an identity claim. The utterances are then processed in a speaker recognition mode and compared with speaker models associated with the identity claim. Speaker models are trained by recording and processing prompted verification phrases in an enrollment session.

In addition to security applications, speaker verification may be used to offer personalized services to users. For example, once a speaker verification phrase is authenticated, the user may be given access to a personalized phone book for voice repertory dialing.

A forensic application is likely to be an open-set identification or verification task. A sample of speech exists from an unknown perpetrator. A suspect is required to speak utterances contained in the suspect speech sample in order to train a model. The suspect speech sample is compared both with the suspect and nonsuspect models to decide whether to accept or reject the hypothesis that the suspect and perpetrator voices are the same.

In surveillance applications the input speech mode is most likely to be text independent. Since the speaker may be unaware that his voice is being monitored, he cannot be expected to speak specified texts. The decision task is open-set identification or verification.

Large amounts of multimedia data, including speech, are being recorded and stored on digital media. The existence of such large amounts of data has created a need for efficient, versatile, and accurate data mining tools for extracting useful information content from the data. A typical need is to search or browse through the data, scanning for specified topics, words, phrase, or speakers. Most of this data is multispeaker data, collected from broadcasts, recorded meetings, telephone conversations, etc. The process of obtaining a list of speaker segments from such data is referred to as speaker indexing, segmentation, or diarization. A more-general task of annotating audio data from various audio sources including speakers has been referred to as audio diarization [36.10].

Still another speaker recognition application is to improve automatic speech recognition by adapting speaker-independent speech models to specified speakers. Many commercial speech recognizers do adapt their speech models to individual users, but this cannot be regarded as a speaker recognition application unless speaker models are constructed and speaker recognition is a part of the process. Speaker recognition can also be used to improve speech recognition for multispeaker data. In this situation speaker indexing can provide a table of speech segments assigned to individual speakers. The speech data in these segments can then be used to adapt speech models to each speaker. Speech recognition of multispeaker speech samples can be improved in another way. Errors and ambiguities in speech recognition transcripts can be corrected using the knowledge provided by speaker segmentation assigning the segments to the correct speakers.

36.2 Measuring Speaker Features

36.2.1 Acoustic Measurements

As mentioned in Sect. 36.1, low-level acoustic features such as short-time spectra are commonly used in speaker modeling. Such features are useful in authentication systems because speakers have less control over spectral details than higher-level features such as pitch.

Short-Time Spectrum

There are many ways of representing the short-time spectrum. A popular representation is the mel-frequency cepstral coefficients (MFCC), which were originally developed for speaker-independent speech recognition. The choice of center frequencies and bandwidths of the filter bank used in MFCC were motivated by the properties of the human auditory system. In particular, this representation provides limited spectral resolution above 2 kHz, which might be detrimental in speaker recognition. However, somewhat counterintuitively, MFCCs have been found to be quite effective in speaker recognition.

There are many minor variations in the definition of MFCC but the essential details are as follows. Let $\{S(k), 0 \le k < K\}$ be the discrete Fourier transform (DFT) coefficients of a windowed speech signal $\hat{s}(t)$. A set of triangular filters are defined such that

$$w_j(k) = \begin{cases} \frac{(k/K) f_s - f_{c_{j-1}}}{f_{c_j} - f_{c_{j-1}}}, & l_j \le k \le c_j, \\ f_{c_{j+1}} - \left(\frac{k}{K}\right) f_s / f_{c_{j+1}} - f_{c_j}, & c_j < k \le u_j, \\ 0, & \text{elsewhere}, \end{cases} \tag{36.1}$$

where $f_{c_{j-1}}$ and $f_{c_{j+1}}$ are the lower and upper limits of the pass band for filter j with $f_{c_0} = 0$ and $f_{c_j} < f_s/2$ for all j, and l_j, c_j and u_j are the DFT indices corresponding to the lower, center, and upper limits of the pass band for filter j. The log-energy at the outputs for the J filters are given by

$$e(j) = \ln\left[\frac{1}{\sum_{k=l_j}^{u_j} w_j(k)} \sum_{k=l_j}^{u_j} \|S(k)\|^2 w_j(k)\right] , \quad (36.2)$$

and the MFCC coefficients are the discrete cosine transform of the filter energies computed as

$$C(k) = \sum_{j=0}^{J} e(j) \cos\left[k\left(j - \frac{1}{2}\right)\frac{\pi}{J}\right] ,$$
$$k = 1, 2, \ldots, K . \quad (36.3)$$

The zeroth coefficient $C(0)$ is set to be the average log-energy of the windowed speech signal. Typical values of the various parameters involved in the MFCC computation are as follows. A cepstrum vector is calculated using a window length of 20 ms and updated every 10 ms. The center frequencies f_{c_j} are uniformly spaced from 0 to 1000 Hz and logarithmically spaced above 1000 Hz. The number of filter energies is typically 24 for telephone-band speech and the number of cepstrum coefficients used in modeling varies from 12 to 18 [36.13].

Cepstral coefficients based on short-time spectra estimated using linear predictive analysis and perceptual linear prediction are other popular representations [36.14].

Short-time spectral measurements are sensitive to channel and transducer variations. Cepstral mean subtraction (CMS) is a simple and effective method to compensate for convolutional distortions introduced by slowly varying channels. In this method, the cepstral vectors are transformed such that they have zero mean. The cepstral average over a sufficiently long speech signal approximates the estimate of a stationary channel [36.14]. Therefore, subtracting the mean from the original vectors is roughly equivalent to normalizing the effects of the channel, if we assume that the average of the clean speech signal is zero. Cepstral variance normalization, which results in feature vectors with unit variance, has also been shown to improve performance in text-independent speaker recognition when there is more than a minute of speech for enrollment. Other feature normalization methods, such as feature warping [36.15] and Gaussianization [36.16], map the observed feature distribution to a normal distribution over a sliding window, and have been shown to be useful in speaker recognition.

It has been long established that incorporating dynamic information is useful for speaker recognition and speech recognition [36.17]. The dynamic information is typically incorporated by extending the static cepstral vectors by their first and second derivatives computed as:

$$\Delta C_k = \frac{\sum_{t=-l}^{l} t c_{t+k}}{\sum_{t=-l}^{l} |t|} , \quad (36.4)$$

$$\Delta\Delta C_k = \frac{\sum_{t=-l}^{l} t^2 c_{t+k}}{\sum_{t=-l}^{l} t^2} . \quad (36.5)$$

Pitch

Voiced sounds are produced by a quasiperiodic opening and closing of the vocal folds in the larynx at a *fundamental frequency* that depends on the speaker. Pitch is a complex auditory attribute of sound that is closely related to this fundamental frequency. In this chapter, the term pitch is used simply to refer to the measure of periodicity observed in voiced speech.

Prosodic information represented by pitch and energy contours has been used successfully to improve the performance of speaker recognition systems [36.18]. There are a number of techniques for estimating pitch from the speech signal [36.19] and the performance of even simple pitch-estimation techniques is adequate for speaker recognition. The major failure modes occur during speech segments that are at the boundaries of voiced and unvoiced sounds and can be ignored for speaker recognition. A more-significant problem with using pitch information for speaker recognition is that speakers have a fair amount of control over it, which results in large intraspeaker variations and mismatch between enrollment and test utterances.

36.2.2 Linguistic Measurements

In traditional speaker authentication applications, the enrollment data is limited to a few repetitions of a password, and the same password is spoken to gain access to the system. In such cases, speaker models based on short-time spectra are very effective and it is difficult to

extract meaningful *high-level* or linguistic features. In applications such as indexing broadcasts by speaker and passive surveillance, a significant amount of enrollment data, perhaps several minutes, may be available. In such cases, the use of linguistic features has been shown to be beneficial [36.18].

Word Usage

Features such as vocabulary choices, function word frequencies, part-of-speech frequencies, etc., have been shown to be useful in speaker recognition [36.20]. In addition to words, spontaneous speech contains fillers and hesitations that can be characterized by statistical models and used for identifying speakers [36.20, 21]. There are a number of issues with speaker recognition systems based on lexical features: they are susceptible to errors introduced by large-vocabulary speech recognizers, a significant amount of enrollment data is needed to build robust models, and the speaker models are likely to characterize the topic of conversation as well as the speaker.

Phone Sequences and Lattices

Models of phone sequences output by speech recognizers using phonotactic grammars, typically phone unigrams, can be used to represent speaker characteristics [36.22]. It is assumed that these models capture speaker-specific pronunciations of frequently occurring words, choice of words, and also an implicit characterization of the acoustic space occupied by the speech signal from a given speaker. It turns out that there is an optimal tradeoff between the constraints used in the recognizer to produce the phone sequences and the robustness of the speaker models of phone sequences. For example, the use of lexical constraints in the automatic speech recognition (ASR) reproduces phone sequences found in a predetermined dictionary and prevents phone sequences that may be characteristic of a speaker but not represented in the dictionary.

The phone accuracy computed using one-best output phone strings generated by ASR systems without lexical constraints is typically not very high. On the other hand, the correct phone sequence can be found in a phone lattice output by an ASR with a high probability. It has been shown that it is advantageous to construct speaker models based on phone-lattice output rather than the one-best phone sequence [36.22]. Systems based on one-best phone sequences use the counts of a term such as a phone unigram or bigram in the decoded sequence. In the case of lattice outputs, these raw counts are replaced by the expected counts given by

$$\boldsymbol{E}[C(\tau|X)] = \sum_{Q} p(Q|X)C(\tau|Q)\,, \tag{36.6}$$

where Q is a path through the phone lattice for the utterance X with associated probability $p(Q|X)$, and $C(\tau|Q)$ is the count of the term τ in the path Q.

Other Linguistic Features

A number of other features that have been found to be useful for speaker modeling are (a) pronunciation modeling of carefully chosen words, and (b) prosodic statistics such as pitch and energy contours as well as durations of phones and pauses [36.23].

36.3 Constructing Speaker Models

A speaker recognition system provides the ability to construct a model λ_s for speaker s using enrollment utterances from that speaker, and a method for comparing the quality of match of a *test* utterance to the speaker model. The choice of models is determined by the application constraints. In applications in which the user is expected to say a fixed password each time, it is beneficial to develop models for words or phrases to capture the temporal characteristics of speech. In passive surveillance applications, the test utterance may contain phonemes or words not seen in the enrollment data. In such cases, less-detailed models that model the overall acoustic space of the user's utterances tend to be effective. A survey of general techniques that have been used in speaker modeling follows. The methods can be broadly classified as nonparametric or parametric. Nonparametric models make few structural assumptions and are effective when there is sufficient enrollment data that is matched to the test data. Parametric models allow a parsimonious representation of the structural constraints and can make effective use of the enrollment data if the constraints are appropriately chosen.

36.3.1 Nonparametric Approaches

Templates

This is the simplest form of speaker modeling and is appropriate for fixed-password speaker verification sys-

tems [36.24]. The enrollment data consists of a small number of repetitions of the password spoken by the target speaker. Each enrollment utterance $\boldsymbol{X}$ is a sequence of feature vectors $\{\boldsymbol{x}_t\}_{t=0}^{T-1}$ generated as described in Sect. 36.2, and serves as the *template* for the password as spoken by the target speaker. A test utterance $\boldsymbol{Y}$ consisting of vectors $\{\boldsymbol{y}_t\}_{t=0}^{T'-1}$, is compared to each of the enrollment utterances and the identity claim is accepted if the distance between the test and enrollment utterances is below a decision threshold. The comparison is done as follows. Associated with each pair of vectors, x_i and y_j, is a distance, $d(\boldsymbol{x}_i, \boldsymbol{y}_j)$. The feature vectors of $\boldsymbol{X}$ and $\boldsymbol{Y}$ are aligned using an algorithm referred to as *dynamic time warping* to minimize an overall distance defined as the average intervector distance $d(x_i, y_j)$ between the aligned vectors [36.12].

This approach is effective in simple fixed-password applications in which robustness to channel and transducer differences are not an issue. This technique is described here mostly for historical reasons and is rarely used in real applications today.

Nearest-Neighbor Modeling

Nearest-neighbor models have been popular in nonparametric classification [36.25]. This approach is often thought of as estimating the local density of each class by a Parzen estimate and assigning the test vector to the class with the maximum local density. The local density of a class (speaker) with enrollment data $\boldsymbol{X}$ at a test vector $\boldsymbol{y}$ is defined as

$$p_{\mathrm{nn}}(\boldsymbol{y}; \boldsymbol{X}) = \frac{1}{V[d_{\mathrm{nn}}(\boldsymbol{y}, \boldsymbol{X})]}\,, \tag{36.7}$$

where $d_{\mathrm{nn}}(\boldsymbol{y}, \boldsymbol{X}) = \min_{\boldsymbol{x}_j \in \boldsymbol{X}} \|\boldsymbol{y} - \boldsymbol{x}_j\|$ is the nearest-neighbor distance and $V(r)$ is the volume of a sphere of radius r in the D-dimensional feature space. Since $V(r)$ is proportional to r^D,

$$\ln[p_{\mathrm{nn}}(\boldsymbol{y}; \boldsymbol{X})] \approx -D \ln[d_{\mathrm{nn}}(\boldsymbol{y}, \boldsymbol{X})]\,. \tag{36.8}$$

The log-likelihood score of the test utterances $\boldsymbol{Y}$ with respect to a speaker specified by enrollment $\boldsymbol{X}$ is given by

$$s_{\mathrm{nn}}(\boldsymbol{Y}; \boldsymbol{X}) \approx -\sum_{\boldsymbol{y}_j \in \boldsymbol{Y}} \ln[d_{\mathrm{nn}}(\boldsymbol{y}, \boldsymbol{X})]\,, \tag{36.9}$$

and the speaker with the greatest $s(\boldsymbol{Y}; \boldsymbol{X})$ is identified.

A modified version of the nearest-neighbor model, motivated by the discussion above, has been successfully used in speaker identification [36.26]. It was found empirically that a score defined as

$$\begin{aligned} s'_{\mathrm{nn}}(\boldsymbol{Y}; \boldsymbol{X}) = {} & \frac{1}{N_y} \sum_{\boldsymbol{y}_j \in \boldsymbol{Y}} \min_{\boldsymbol{x}_i \in \boldsymbol{X}} \|\boldsymbol{y}_j - \boldsymbol{x}_i\|^2 \\ & + \frac{1}{N_x} \sum_{\boldsymbol{x}_j \in \boldsymbol{X}} \min_{\boldsymbol{y}_i \in \boldsymbol{Y}} \|\boldsymbol{y}_i - \boldsymbol{x}_j\|^2 \\ & - \frac{1}{N_y} \sum_{\boldsymbol{y}_j \in \boldsymbol{Y}} \min_{\boldsymbol{y}_i \in \boldsymbol{Y};\, j \neq i} \|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2 \\ & - \frac{1}{N_x} \sum_{\boldsymbol{x}_j \in \boldsymbol{X}} \min_{\boldsymbol{x}_i \in \boldsymbol{X};\, j \neq i} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 \end{aligned} \tag{36.10}$$

gives much better performance than $s_{\mathrm{nn}}(\boldsymbol{Y}; \boldsymbol{X})$.

36.3.2 Parametric Approaches

Vector Quantization Modeling

Vector quantization constructs a set of representative samples of the target speaker's enrollment utterances by clustering the feature vectors. Although a variety of clustering techniques exist, the most commonly used is k-means clustering [36.14]. This approach partitions N feature vectors into K disjoint subsets S_j to minimize an overall distance such as

$$D = \sum_{j=1}^{J} \sum_{\boldsymbol{x}_i \in S_j} (\boldsymbol{x}_i - \boldsymbol{\mu}_j)\,, \tag{36.11}$$

where $\boldsymbol{\mu}_j = (1/N_j) \sum_{\boldsymbol{x}_i \in S_j} \boldsymbol{x}_i$ is the centroid of the N_j samples in the j-th cluster. The algorithm proceeds in two steps:

1. Compute the centroid of each cluster using an initial assignment of the feature vectors to the clusters.
2. Reassign $\boldsymbol{x}_i$ to that cluster whose centroid is closest to it.

These steps are iterated until successive steps do not reassign samples.

This algorithm assumes that there exists an initial clustering of the samples into K clusters. It is difficult to obtain a good initialization of K clusters in one step. In fact, it may not even be possible to reliably estimate K clusters because of data sparsity. The Linde–Buzo–Gray (LBG) algorithm [36.27] provides a good solution for this problem. Given m centroids, the LBG algorithm produces additional centroids by perturbing one or more of the centroids using a heuristic. One common heuristic is to choose the $\boldsymbol{\mu}$ for the cluster with the largest variance and produce two centroids $\boldsymbol{\mu}$ and $\boldsymbol{\mu} + \boldsymbol{\epsilon}$. The enrollment feature vectors are assigned to the resulting $m+1$ centroids. The k-means algorithm described previously can

then be applied to refine the centroid estimates. This process can be repeated until $m = M$ or the cluster sizes fall below a threshold. The LBG algorithm is usually initialized with $m = 1$ and computes the centroid of all the enrollment data. There are many variations of this algorithm that differ in the heuristic used for perturbing the centroids, the termination criteria, and similar details. In general, this algorithm for generating VQ models has been shown to be quite effective. The choice of K is a function of the size of enrollment data set, the application, and other system considerations such as limits on computation and memory.

Once the VQ models are established for a target speaker, scoring consists of evaluating D in (36.11) for feature vectors in the test utterance. This approach is general and can be used for text-dependent and text-independent speaker recognition, and has been shown to be quite effective [36.28]. Vector quantization models can also be constructed on sequences of feature vectors, which are effective at modeling the temporal structure of speech. If distance functions and centroids are suitably redefined, the algorithms described in this section continue to be applicable.

Although VQ models are still useful in some situations, they have been superseded by models such as the Gaussian mixture models and hidden Markov models, which are described in the following sections.

Gaussian Mixture Models

In the case of text-independent speaker recognition (the subject of Chap. 38) where the system has no prior knowledge of the text of the speaker's utterance, Gaussian mixture models (GMMs) have proven to be very effective. This can be thought of as a refinement of the VQ model. Feature vectors of the enrollment utterances $\boldsymbol{X}$ are assumed to be drawn from a probability density function that is a mixture of Gaussians given by

$$p(\boldsymbol{x}|\lambda) = \sum_{k=1}^{K} w_k p_k(\boldsymbol{x}|\lambda_k) \,, \tag{36.12}$$

where $0 \le w_k \le 1$ for $1 \le k \le K$, $\sum_{k=1}^{K} w_k = 1$, and

$$p_k(\boldsymbol{x}|\lambda_k) = \frac{1}{(2\pi)^{D/2}|\Sigma_k|^{1/2}} \times \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^{\mathrm{T}}\Sigma_k^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k)\right) \,, \tag{36.13}$$

λ represents the parameters $(\boldsymbol{\mu}_i, \Sigma_i, w_i)_{i=1}^{K}$ of the distribution. Since the size of the training data is often small, it is difficult to estimate full covariance matrices reliably. In practice, $\{\Sigma_k\}_{k=1}^{K}$ are assumed to be diagonal.

Given the enrollment data $\boldsymbol{X}$, the maximum-likelihood estimates of the λ can be obtained using the *expectation-maximization* (EM) algorithm [36.12]. The K-means algorithm can be used to initialize the parameters of the component densities. The posterior probability that $\boldsymbol{x}_t$ is drawn from the component $p_m(\boldsymbol{x}_t|\lambda_m)$ can be written

$$P(m|\boldsymbol{x}_t, \lambda) = \frac{w_m p_m(\boldsymbol{x}_t|\lambda_m)}{p(\boldsymbol{x}_t|\lambda)} \,. \tag{36.14}$$

The maximum-likelihood estimates of the parameters of λ in terms of $P(m|\boldsymbol{x}_t, \lambda)$ are

$$\boldsymbol{\mu}_m = \frac{\sum_{t=1}^{\mathrm{T}} P(m|\boldsymbol{x}_t, \lambda)\boldsymbol{x}_t}{\sum_{t=1}^{\mathrm{T}} P(m|\boldsymbol{x}_t, \lambda)} \,, \tag{36.15}$$

$$\Sigma_m = \frac{\sum_{t=1}^{\mathrm{T}} P(m|\boldsymbol{x}_t, \lambda)\boldsymbol{x}_t\boldsymbol{x}_t^{\mathrm{T}}}{\sum_{t=1}^{\mathrm{T}} P(m|\boldsymbol{x}_t, \lambda)} - \boldsymbol{\mu}_m\boldsymbol{\mu}_m^{\mathrm{T}} \,, \tag{36.16}$$

$$w_m = \frac{1}{T}\sum_{t=1}^{\mathrm{T}} P(m|\boldsymbol{x}_t, \lambda) \,. \tag{36.17}$$

The two steps of the EM algorithm consist of computing $P(m|\boldsymbol{x}_t, \lambda)$ given the current model, and updating the model using the equations above. These two steps are iterated until a convergence criteria is satisfied.

Test utterance scores are obtained as the average log-likelihood given by

$$s(\boldsymbol{Y}|\lambda) = \frac{1}{T}\sum_{t=1}^{\mathrm{T}} \log[p(\boldsymbol{y}_t|\lambda)] \,. \tag{36.18}$$

Speaker verification is often based on a likelihood-ratio test statistic of the form $p(\boldsymbol{Y}|\lambda)/p(\boldsymbol{Y}|\lambda_{\mathrm{bg}})$ where λ is the speaker model and λ_{bg} represents a background model [36.29]. For such systems, speaker models can also be trained by adapting λ_{bg}, which is generally trained on a large independent speech database [36.30]. There are many motivations for this approach. Generating a speaker model by adapting a well-trained background GMM may yield models that are more robust to channel differences, and other kinds of mismatch between enrollment and test conditions than models estimated using only limited enrollment data. Details of this procedure can be found in Chap. 38.

Speaker modeling using GMMs is attractive for text-independent speaker recognition because it is simple to implement and computationally inexpensive. The fact that this model does not model temporal aspects of speech is a disadvantage. However, it has been difficult to exploit temporal structure to improve speaker recognition performance when the linguistic content of test utterances does not overlap significantly with the linguistic content of enrollment utterances.

Hidden Markov Models

In applications where the system has prior knowledge of the text and there is significant overlap of what was said during enrollment and testing, text-dependent statistical models are much more effective than GMMs. An example of such applications is access control to personal information or bank accounts using a voice password. Hidden Markov models (HMMs) [36.12] for phones, words, or phrases, have been shown to be very effective [36.31, 32]. Passwords consisting of word sequences drawn from specialized vocabularies such as digits are commonly used. Each word can be characterized by an HMM with a small number of states, in which each state is represented by a Gaussian mixture density. The maximum-likelihood estimates of the parameters of the model can be obtained using a generalization of the EM algorithm [36.12].

The ML training aims to approximate the underlying distribution of the enrollment data for a speaker. The estimates deviate from the *true* distribution due to lack of sufficient training data and incorrect modeling assumptions. This leads to a suboptimal classifier design. Some limitations of ML training can be overcome using discriminative training of speaker models in which an attempt is made to minimize an overall cost function that depends on misclassification or detection errors [36.33–35]. Discriminative training approaches require examples from competing speakers in addition to examples from the target speaker. In the case of closed-set speaker identification, it is possible to construct a misclassification measure to evaluate how likely a test sample, spoken by a target speaker, is misclassified as any of the others. One example of such a measure is the *minimum classification error* (MCE) defined as follows. Consider the set of S discriminant functions $\{g_k(\boldsymbol{x}; \Lambda_s), 1 \le s \le S\}$, where $g_k(\boldsymbol{x}; \Lambda_s)$ is the log-likelihood of observation $\boldsymbol{x}$ given the models Λ_s for speaker s. A set of misclassification measures for each speaker can be defined as

$$d_s(\boldsymbol{x}; \Lambda) = -g_s(\boldsymbol{x}; \Lambda_s) + G_s(\boldsymbol{x}; \Lambda), \tag{36.19}$$

where Λ is the set of all speaker models and $G_s(\boldsymbol{x}; \Lambda)$ is the antidiscriminant function for speaker s. $G_s(\boldsymbol{x}; \Lambda)$ is defined so that $d_s(\boldsymbol{x}; \Lambda)$ is positive only if $\boldsymbol{x}$ is incorrectly classified. In speech recognition problems, $G_s(\boldsymbol{x}; \Lambda)$ is usually defined as a collective representation of all competing classes. In the speaker identification task, it is often advantageous to construct pairwise misclassification measures such as

$$d_{ss'}(\boldsymbol{x}; \Lambda) = -g_s(\boldsymbol{x}; \Lambda_s) + g'_s\left(\boldsymbol{x}; \Lambda'_s\right), \tag{36.20}$$

with respect to a set of competing speakers s', a subset of the S speakers. Each misclassification measure is embedded into a smooth empirical *loss function*

$$l_{ss'}(\boldsymbol{x}; \Lambda) = \frac{1}{1 + \exp(-\alpha d_{ss'}(\boldsymbol{x}; \Lambda))}, \tag{36.21}$$

which approximates a loss directly related to the number of classification errors, and α is a smoothness parameter. The loss functions can then be combined into an overall loss given by

$$l(\boldsymbol{x}; \Lambda) = \sum_s \sum_{s' \in \mathcal{S}_c} l_{ss'}(\boldsymbol{x}; \Lambda) \delta_s(\boldsymbol{x}), \tag{36.22}$$

where $\delta_s(\boldsymbol{x})$ is an indicator function which is equal to 1 when $\boldsymbol{x}$ is uttered by speaker s and 0 otherwise, and $\mathcal{S}_c$ is the set of competing speakers. The total loss, defined as the sum of $l(\boldsymbol{x}; \Lambda)$ over all training data, can be optimized with respect to all the model parameters using a gradient-descent algorithm. A similar algorithm has been developed for speaker verification in which samples from a large number of speakers in a development set is used to compute a minimum verification measure [36.36].

The algorithm described above is only to illustrate the basic principles of discriminative training for speaker identification. Many other approaches that differ in their choice of the loss function or the optimization method have been developed and shown to be effective [36.35, 37].

The use of HMMs in text-dependent speaker verification is discussed in detail in Chap. 37.

Support Vector Modeling

Traditional discriminative training approaches such as those based on MCE have a tendency to overtrain on the training set. The complexity and generalization ability of the models are usually controlled by testing on

a held-out development set. Support vector machines (SVMs) [36.38] provide a way for training classifiers using discriminative criteria and in which the model complexity that provides good generalization to test data is determined automatically from the training data. SVMs have been found to be useful in many classification tasks including speaker identification [36.39].

The original formulation of SVMs was for two-class problems. This seems appropriate for speaker verification in which the positive samples consist of the enrollment data from a target user and the negative samples are drawn from a large set of imposter speakers. Many extensions of SVMs to multiclass classification have also been developed and are appropriate for speaker identification. There are many issues with SVM modeling for speaker recognition, including the appropriate choice of features and the kernel. The use of SVMs for text-independent speaker recognition is the subject of Chap. 38.

Other Approaches

Most state-of-the-art speaker recognition systems use some combination of the modeling methods described in the previous sections. Many other interesting models have been proposed and have been shown to be useful in limited scenarios. Eigenvoice modeling is an approach in which the speaker models are confined to a low-dimensional linear subspace obtained using independent training data from a large set of speakers. This method has been shown to be effective for speaker modeling and speaker adaptation when the enrollment data is too limited for the effective use of other text-independent approaches such as GMMs [36.40]. Artificial neural networks [36.41] have also been shown to be useful in some situations, perhaps in combination with GMMs. When sufficient enrollment data is available, a method for speaker detection that involves comparing the test segment directly to similar segments in enrollment data has been shown to be effective [36.42].

36.4 Adaptation

In most speaker recognition scenarios, the speech data available for enrollment is too limited for training models that adequately characterize the range of test conditions in which the system needs to operate. For example, in fixed-password speaker authentication systems used in telephony services, enrollment data is typically collected in a single call. The enrollment and test conditions may be mismatched in a number of ways: the telephone handset that is used, the location of the call, which determines the kinds of background noises, and the channel over which speech is transmitted such as cellular or landline networks. In text-independent modeling, there are likely to be additional problems because of mismatch in the linguistic content. A very effective way to mitigate the effects of mismatch is model adaptation.

Models can be adapted in an unsupervised way using data from authenticated utterances. This is common in fixed-password systems and can reduce the error rate significantly. It is also necessary to update the decision thresholds when the models are adapted. Since the selection of data for model adaptation is not supervised, there is the possibility that models are adapted on imposter utterances. This can be disastrous. The details of unsupervised model and threshold adaptation and the various issues involved are explained in detail in Chap. 37.

Speaker recognition is often incorporated into other applications that involve a dialog with the user. Feedback from the dialog system can be used to supervise model adaptation. In addition, meta-information available from a dialog system such as the history of interactions can be combined with speaker recognition to design a flexible and secure authentication system [36.43].

36.5 Decision and Performance

36.5.1 Decision Rules

Whether they are used for speaker identification or verification, the various models and approaches presented in Sect. 36.3 provide a score $s(Y|\lambda)$ measuring the match between a given test utterance Y and a speaker model λ. Identification systems yield a set of such scores corresponding to each speaker in a target list. Verification systems output only one score using the speaker model of the claimed speaker. An

accept or reject decision has to be made using this score.

Decision in closed-set identification consists of choosing the identified speaker $\hat{S}$ as the one that corresponds to the maximum score:

$$\hat{S} = \arg\max_j s(Y|\lambda_j) \,, \tag{36.23}$$

where the index j ranges over the whole set of target speakers.

Decision in verification is obtained by comparing the score computed using the model for the claimed speaker S_i given by $s(Y|\lambda_i)$ to a predefined threshold θ. The claim is accepted if $s(Y|\lambda_i) \geq \theta$, and rejected otherwise.

Open-set identification relies on a step of closed-set identification eliciting the most likely identity, followed by a verification step to determine whether the hypothesized identity match is good enough.

36.5.2 Threshold Setting and Score Normalization

Efficiency and robustness require that the score $s(Y|\lambda)$ be quite readily exploited in a practical application. In particular, the threshold θ should be as insensitive as possible across users and application context.

When the score is obtained in a probabilistic framework or can be interpreted as a (log) likelihood ratio (LLR), Bayesian decision theory [36.44] states that an optimal threshold for verification can be theoretically set once the desired false acceptance c_{fa} and false rejection c_{fr}, and the a priori probability p_{imp} of an impostor trying to enter the system, are specified. The optimal choice of the threshold is given by:

$$\theta^* = \frac{c_{\text{fa}}}{c_{\text{fr}}} \frac{p_{\text{imp}}}{1 - p_{\text{imp}}} \,. \tag{36.24}$$

In practice, however, the score $s(Y|\lambda)$ does not behave as theory would predict since the statistical models are not ideal. Various normalization procedures have been proposed to alleviate this problem. Initial work by *Li* and *Porter* [36.45] has inspired a number of score normalization techniques that intend to make the statistical distribution of $s(Y|\lambda)$ as independent as possible across speakers, acoustic conditions, linguistic content, etc. This has lead to a number of threshold normalization schemes, such as the Z-norm, H-Norm, and T-norm, which use side information, the distance between models, and speech material from a development set to determine the normalization parameters. These normalization procedures are discussed in more detail in Chaps. 37, 38 and [36.46]. Even so, the optimal threshold for a given operating condition is generally estimated experimentally from development data that is appropriate for a given scenario.

36.5.3 Errors and DET Curves

The performance of an identification system is related to the probability of misclassification, which corresponds to cases when the identified speaker is not the actual one.

Verification systems are evaluated based on two types of errors: false acceptance, when an impostor speaker succeeds in being verified with an erroneous claimed identity, and false rejection, when a target user claiming his/her genuine identity is rejected. The *a posteriori* estimates of the probabilities p_{fa} and p_{fr} of these two types of errors vary in the opposite way from each other when the decision threshold θ is varied. The tradeoff between p_{fa} and p_{fr} (sometimes mapped to the probability of detection p_{d}, defined as $1 - p_{\text{fr}}$) is often displayed in the form of a receiver operating characteristic (ROC), a term commonly used in detection theory [36.44]. In speaker recognition systems a different representation of the same data, referred to as the detection error tradeoff (DET) curve, has become popular.

The DET curve [36.47] is the standard way to depict the system behavior in terms of hypotheses separability by plotting p_{fa} as a function of p_{fr}. Rather than the probabilities themselves, the normal deviates corresponding to the probabilities are plotted. For a particular threshold value, the corresponding error rates p_{fa} and p_{fr} appear as a specific point on this DET curve. A popular point is the one where $p_{\text{fa}} = p_{\text{fr}}$, which is called the equal error rate (EER). Plotting DET curves is a good way to compare the potential of two methods in a laboratory but it is not suited for predicting accurately the performance of a system when deployed in real-life conditions.

The decision threshold θ is often chosen to optimize a cost that is a function of the probability of false acceptance and false rejection as well as the prior probability of an imposter attack. One such function is called the detection cost function (DCF), defined as [36.48]

$$C = p_{\text{imp}} c_{\text{fa}} p_{\text{fa}} + (1 - p_{\text{imp}}) c_{\text{fr}} p_{\text{fr}} \,. \tag{36.25}$$

The DCF is indeed a way to evaluate a system under a particular operating condition and to summarize into a single figure its estimated performance in a given application scenario. It has been used as the primary figure of merit for the evaluation of systems participating in the yearly NIST speaker recognition evaluations [36.48].

36.6 Selected Applications for Automatic Speaker Recognition

Text-dependent and text-independent speaker recognition technology and their applications are discussed in detail in the following two Chaps. 37 and 38. A few interesting, but perhaps not primary, applications of speaker recognition technology are described in this section. These applications were chosen to demonstrate the wide range of applications of speaker recognition.

36.6.1 Indexing Multispeaker Data

Speaker indexing can be approached as either a supervised or unsupervised task. Supervised means that prior speaker models exist for the speakers of interest included in the data. The data can then be scanned and processed to determine the segments associated with each of these speakers. Unsupervised means that prior speaker models do not exist. The type of approach taken depends on the type and amount of prior knowledge available for particular applications. There may be knowledge of the identities of the participating speakers and there may even be independent labeled speech data available for constructing models for these speakers, such as in the case of some broadcast news applications [36.6, 49, 50]. In this situation the task is supervised and the techniques for speaker segmentation or indexing are basically the same as used for speaker detection [36.9, 50, 51].

A more-challenging task is unsupervised segmentation. An example application is the segmentation of the speakers in a two-person telephone conversation [36.4, 9, 52, 53]. The speaker identities may or may not be known but independent labelled speech data for constructing speaker models is generally not available. Following is a possible approach to the unsupervised segmentation problem. The first task is to construct unlabeled single-speaker models from the current data. An initial segmentation of the data is carried out with an acoustic change detector using a criterion such as the generalized likelihood ratio (GLR) [36.4, 5] or Bayesian information criterion (BIC) [36.8, 54, 55]. The hypothesis underlying this process is that each of the resulting segments will be single-speaker segments. These segments are then clustered using an agglomerative clustering algorithm with a criterion for measuring the pairwise similarity between segments [36.56–58]. Since in the cited application the number of speakers is known to be two, the clustering terminates when two clusters are obtained. If the acoustic change criterion and the matching criterion for the clustering perform well the two clusters of segments will each contain segments mostly from one speaker or the other. These segment clusters can then be used to construct protospeaker models, typically GMMs. Each of these models is then used to resegment the data to provide an improved segmentation which, in turn, will provide improved speaker models. The process can be iterated until no further significant improvement is obtained. It then remains to apply speaker labels to the models and segmentations. Some independent knowledge is required to accomplish this. As mentioned earlier, the speakers in the telephone conversation may be known, but some additional information is required to assign labels to the correct models and segmentations.

36.6.2 Forensics

The perspective of being able to identify a person on the basis of his or her voice has received significant interest in the context of law enforcement. In many situations, a voice recording is a key element, and sometimes the only one available, for proceeding with an investigation, identifying or clearing a suspect, and even supporting an accusation or defense in a court of law.

The public perception is that voice identification is a straightforward task, and that there exists a reliable *voiceprint* in much the same way as there are fingerprints or genetic (DNA) prints. This is not true in general because the voice of an individual has a strong behavioral component, and is only partly based on anatomical properties. Moreover, the conditions under which the test utterance is recorded are generally not known or controlled. The test voice sample might be from an anonymous call, wiretapping, etc. For these reasons, the use of voice recognition in the context of forensic applications must be approached with caution [36.59].

The four procedures that are generally followed in the forensic context are described below.

Nonexpert Speaker Recognition by Lay Listener(s)

This procedure is used in the context of a voice lineup when a victim or a witness has had the opportunity of hearing a voice sample and is asked to say whether he or she recognizes this voice, or to determine if this voice sample matches one of a set of utterances. Since it is difficult to set up such a test in a controlled way and calibrate to the matching criteria an individual subject may use, such procedures can be used only to suggest a possible course of action during an investigation.

Expert Speaker Recognition

Expert study of a voice sample might include one or more of aural–perceptual approaches, linguistic analysis, and spectrogram examination. In this context, the expert takes into account several levels of speaker characterization such as pitch, timbre, diction, style, idiolect, and other idiosyncracies, as well as a number of physical measurements including fundamental frequencies, segment durations, formants, and jitter. Experts provide a decision on a seven-level scale specified by the International Association for Identification (IAI) standard [36.60] on whether two voice samples (the disputed recording and a voice sample of the suspect) are more or less likely to have been produced by a the same person. Subjective heterogeneous approaches coexist between forensic practitioners and, although the technical invalidity of some methods has been clearly established, they are still used by some. The expert-based approach is therefore generally used with extreme caution.

Semiautomatic Methods

This category refers to systems for which a supervised selection of speech segments is conducted prior to a computer-based analysis of the selected material. Whereas a calibrated metric can be used to evaluate the similarity of specific types of segments such as words or phrases, these systems tend to suffer from a lack of standardization.

Automatic Methods

Fully automated methods using state-of-the-art techniques offer an attractive paradigm for forensic speaker verification. In particular, these automatic approaches can be run without any (subjective) human intervention, they offer a reproducible procedure, and they lend themselves to large-scale evaluation. Technological improvements over the years, as well as progress in the presentation, reporting, and interpretation of the results, have made such methods attractive. However, levels of performance remain highly sensitive to a number of external factors ranging from the quality and similarity of recording conditions, the cooperativeness of speakers, and the potential use of technologies to fake or disguise a voice.

Thanks to a number of initiatives and workshops (in particular the series of ISCA and IEEE Odyssey workshops), the past decade has seen some convergence in terms of formalism, interpretation, and methodology between forensic science and engineering communities. In particular, the interpretation of voice forensic evidence in terms of Bayesian decision theory and the growing awareness of the need for systematic evaluation have constituted significant contributions to these exchanges.

36.6.3 Customization: SCANmail

Customization of services and applications to the user is another class of applications of speaker recognition technology. An example of a customized messaging system is one where members of a family share a voice mailbox. Once the family members are enrolled in a speaker recognition system, there is no need for them to identify themselves when accessing their voice mail. A command such as *Get my messages* spoken by a user can be used to identify and authenticate the user, and provide only those messages left for that user. There are many such applications of speaker recognition technology. An interesting and successful application of *caller identification* to a voicemail browser is described in this section.

SCANMail is a system developed for the purpose of providing useful tools for managing and searching through voicemail messages [36.61]. It employs ASR to provide text transcriptions, information retrieval on the transcriptions to provide a weighted set of search terms, information extraction to obtain key information such as telephone numbers from transcription, as well as automatic speaker recognition to carry out caller identification by processing the incoming messages. A graphical user interface enables the user to exercise the features of the system. The caller identification function is described in more detail below.

Two types of processing requests are handled by the caller identification system (CIS). The first type of request is to assign a speaker label to an incoming message. When a new message arrives, ASR is used to produce a transcription. The transcription as well as the speech signal is transmitted to the CIS for caller identification. The CIS compares the processed speech signal with the model of each caller in the recipient's address book. The recipient's address book is populated with speaker models when the user adds a caller to the address book by providing a label to a received message. A matching score is obtained for each of the caller models and compared to a caller-dependent rejection threshold. If the matching score exceeds the threshold, the received message is assigned a speaker label. Otherwise, CIS assigns an *unknown* label to the message.

The second type of request originates with the user action of adding a caller to an address book as mentioned earlier. In the course of reviewing a received message, the user has the capability to supply a caller label to the

message. The enrollment module in the CIS attempts to construct a speaker model for a new user using that message. The acoustic models are trained using text-independent speaker modeling. Acoustic models can be augmented with models based on meta-information, which may include personal information such as the caller's name or contact information left in the message, or the calling history.

36.7 Summary

Identifying speakers by voice was originally investigated for applications in speaker authentication. Over the last decade, the field of speaker recognition has become much more diverse and has found numerous applications. An overview of the technology and sample applications were presented in this chapter.

The modeling techniques that are applicable, and the nature of the problems, vary depending on the application scenario. An important dichotomy is based on whether the content (text) of the speech during training and testing overlaps significantly and is known to the system. These two important cases are the subject of the next two chapters.

References

36.1 J.S. Dunn, F. Podio: Biometrics Consortium website, http://www.biometrics.org (2007)

36.2 M.A. Przybocki, A.F. Martin: The 1999 NIST speaker recognition evaluation, using summed two-channel telephone data for speaker detection and speaker tracking, Eurospeech 1999 Proceedings (1999) pp. 2215–2218, http://www.nist.gov/speech/publications/index.htm

36.3 M.A. Przybocki, A.F. Martin: Nist speaker recognition evaluation chronicles, Odyssey Workshop 2004 Proc. (2004) pp. 15–22

36.4 H. Gish, M.-H. Siu, R. Rohlicek: Segregation of speakers for speech recognition and speaker identification, Proc. ICASSP (1991) pp. 873–876

36.5 L. Wilcox, F. Chen, D. Kimber, V. Balasubramanian: Segmentation of speech using speaker identification, Proc. ICASSP (1994) pp. 161–164

36.6 J.-L. Gauvain, L. Lamel, G. Adda: Partitioning and transcription of broadcast news data, Proc. of ICSLP (1998) pp. 1335–1338

36.7 S.E. Johnson: Who spoke when? – automatic segmentation and clustering for determining speaker turns, Proc. Eurospeech (1999) pp. 2211–2214

36.8 P. Delacourt, C.J. Wellekens: Distbic: A speaker-based segmentation for audio data indexing, Speech Commun. **32**, 111–126 (2000)

36.9 R.B. Dunn, D.A. Reynolds, T.F. Quatieri: Approaches to speaker detection and tracking in conversational speech, Digital Signal Process. **10**, 93–112 (2000)

36.10 S.E. Tranter, D.A. Reynolds: An overview of automatic speaker diarization systems, IEEE Trans. Speech Audio Process. **14**, 1557–1565 (2006)

36.11 L.H. Jamieson: Course notes for speech processing by computer, http://cobweb.ecn.purdue.edu ee649/notes/ (2007) Chap. 1

36.12 L.R. Rabiner, B.-H. Juang: *Fundamentals of Speech Recognition* (Prentice-Hall, Englewood Cliffs 1993)

36.13 S. Davis, P. Mermelstein: Comparison of parametric representation for monosyllable word recognition in continuously spoken sentences, IEEE Trans. Acoust. Speech Signal Process. **28**, 357–366 (1980)

36.14 X. Huang, A. Acero, H.-W. Hon: *Spoken Language Processing: A Guide to Theory, Algorithm and System Development* (Prentice-Hall, Englewood Cliffs 2001)

36.15 J. Pelecanos, S. Sridharan: Feature warping for robust speaker verification, Proc. ISCA Workshop on Speaker Recognition – 2001: A Speaker Odyssey (2001)

36.16 B. Xiang, U. Chaudhari, J. Navratil, G. Ramaswamy, R. Gopinath: Short-time Gaussianization for robust speaker verification, Proc. ICASSP, Vol. 1 (2002) pp. 681–684

36.17 S. Furui: Comparison of speaker recognition methods using static features and dynamic features, IEEE Trans. Acoust. Speech Signal Process. **29**, 342–350 (1981)

36.18 J.P. Campbell, D.A. Reynolds, R.B. Dunn: Fusing high- and low-level features for speaker recognition, Proc. Eurospeech, Vol. 1 (2003)

36.19 W. Hess: *Pitch Determination of Speech Signals* (Springer, Berlin, Heidelberg 1983)

36.20 G. Doddington: Speaker recognition based on idiolectal differences between speakers, Proc. Eurospeech (2001) pp. 2521–2524

36.21 W.D. Andrews, M.A. Kohler, J.P. Campbell, J.J. Godfrey: Phonetic, idiolectal, and acoustic speaker recognition, Proceedings of Odyssey Workshop (2001)

36.22 A. Hatch, B. Peskin, A. Stolcke: Improved phonetic speaker recognition using lattice decoding, Proc. ICASSP, Vol. 1 (2005)

36.23 D. Reynolds, W. Andrews, J. Campbell, J. Navratil, B. Peskin, A. Adami, Q. Jin, D. Klusacek, J. Abramson, R. Mihaescu, J. Godfrey, D. Jones, B. Xiang: The SuperSID project: Exploiting high-level information for high-accuracy speaker recognition, Proc. ICASSP (2003) pp. 784–787

36.24 A.E. Rosenberg: Automatic speaker verification: A review, Proc. IEEE **64**, 475–487 (1976)

36.25 K. Fukunaga: *Introduction to Statistical Pattern Recognition*, 2nd edn. (Elsevier, New York 1990)

36.26 A.L. Higgins, L.G. Bahler, J.E. Porter: Voice identification using nearest-neighbor distance measure, Proc. ICASSP (1993) pp. 375–378

36.27 Y. Linde, A. Buzo, R.M. Gray: An algorithm for vector quantization, IEEE Trans. Commun. **28**, 94–95 (1980)

36.28 F.K. Soong, A.E. Rosenberg, L.R. Rabiner, B.H. Juang: A vector quantization approach to speaker recognition, Proc. IEEE ICASSP (1985) pp. 387–390

36.29 D.A. Reynolds, R.C. Rose: Robust text independent speaker identification using Gaussian mixture speaker models, IEEE Trans. Speech Audio Process. **3**, 72–83 (1995)

36.30 D.A. Reynolds, T.F. Quatieri, R.B. Dunn: Speaker verification using adapted Gaussian mixture models, Digital Signal Process. **10**, 19–41 (2000)

36.31 A.E. Rosenberg, S. Parthasarathy: Speaker background models for connected digit password speaker verification, Proc. ICASSP (1996) pp. 81–84

36.32 S. Parthasarathy, A.E. Rosenberg: General phrase speaker verification using sub-word background models and likelihood-ratio scoring, Proc. Int. Conf. Spoken Language Processing (1996) pp. 2403–2406

36.33 O. Siohan, A.E. Rosenberg, S. Parthasarathy: Speaker identification using minimum classification error training, Proc. ICASSP (1998) pp. 109–112

36.34 A.E. Rosenberg, O. Siohan, S. Parthasarathy: Small group speaker identification with common password phrases, Speech Commun. **31**, 131–140 (2000)

36.35 L. Heck, Y. Konig: Discriminative training of minimum cost speaker verification systems, Proc. RLA2C – Speaker Recognition Workshop (1998) pp. 93–96

36.36 A. Rosenberg, O. Siohan, S. Parthasarathy: Speaker verification using minimum verification error training, Proc. ICASSP (1998) pp. 105–108

36.37 J. Navratil, G. Ramaswamy: Detac – a discriminative criterion for speaker verification, Proc. Int. Conf. Spoken Language Processing (2002)

36.38 V.N. Vapnik: *The Nature of Statistical Learning Theory* (Springer, New York 1995)

36.39 W.M. Campbell, D.A. Reynolds, J.P. Campbell: Fusing discriminative and generative methods for speaker recognition: experiments on switchboard and NFI/TNO field data, Proc. ODYSSEY 2004 – The Speaker and Language Recognition Workshop (2004) pp. 41–44

36.40 O. Thyes, R. Kuhn, P. Nguyen, J.-C. Junqua: Speaker identification and verification using eigenvoices, Proc. ICASSP (2000) pp. 242–245

36.41 K.R. Farrell, R. Mammone, K. Assaleh: Speaker recognition using neural networks and conventional classifiers, IEEE Trans. Speech Audio Process. **2**, 194–205 (1994)

36.42 D. Gillick, S. Stafford, B. Peskin: Speaker detection without models, Proc. ICASSP (2005)

36.43 G.N. Ramaswamy, R.D. Zilca, O. Alecksandrovich: A programmable policy manager for conversational biometrics, Proc. Eurospeech (2003)

36.44 H.V. Poor: *An Introduction to Signal Detection and Estimation* (Springer, Berlin, Heidelberg 1994)

36.45 K.P. Li, J.E. Porter: Normalizations and selection of speech segments for speaker recognition scoring, Proc. IEEE ICASSP (1988) pp. 595–598

36.46 F. Bimbot: A tutorial on text-independent speaker verification, EURASIP J. Appl. Signal Process. **4**, 430–451 (2004)

36.47 A. Martin, G. Doddington, T. Kamm, M. Ordowski, M. Przybocki: The det curve in assessment of detection task performance, Proc. Eurospeech (1997) pp. 1895–1898

36.48 A. Martin, M. Przybocki: The NIST 1999 speaker recognition evaluation – an overview, Digital Signal Process. **10**, 1–18 (2000)

36.49 M.A. Siegler, U. Jain, B. Raj, R.M. Stern: Automatic segmentation, classification, and clustering of broadcast news data, Proc. DARPA Speech Recognition Workshop (1997) pp. 97–99

36.50 A.E. Rosenberg, I. Magrin-Chagnolleau, S. Parthasarathy, Q. Huang: Speaker detection in broadcast news databases, Proc. Int. Conf. on Spoken Lang. Processing (1998) pp. 1339–1342

36.51 J.-F. Bonastre, P. Delacourt, C. Fredouille, T. Merlin, C. Wellekens: A speaker tracking system based on speaker turn detection for nist evaluation, Proc. ICASSP (2000) pp. 1177–1180

36.52 A.G. Adami, S.S. Kajarekar, H. Hermansky: A new speaker change detection method for two-speaker segmentation, Proc. ICASSP (2002) pp. 3908–3911

36.53 A.E. Rosenberg, A. Gorin, Z. Liu, S. Parthasarathy: Unsupervised segmentation of telephone conversations, Proc. Int. Conf. on Spoken Lang. Processing (2002) pp. 565–568

36.54 S.S. Chen, P.S. Gopalakrishnan: Speaker, environment and channel change detection and

clustering via the bayesian information criterion, Proc. DARPA Broadcast News Transcription and Understanding Workshop (1998), http://www.nist.gov/speech/publications/darpa98/index.htm

36.55 A. Tritschler, R. Gopinath: Improved speaker segmentation and segments clustering using the bayesian information criterion, Proc. Eurospeech (1999)

36.56 A.D. Gordon: *Classification: Methods for the Exploratory Analysis of Multivariate Data* (Chapman Hall, Englewood Cliffs 1981)

36.57 F. Kubala, H. Jin, R. Schwartz: Automatic speaker clustering, Proc. DARPA Speech Recognition Workshop (1997) pp. 108–111

36.58 D. Liu, F. Kubala: Online speaker clustering, Proc. ICASSP (2003) pp. 572–575

36.59 J.-F. Bonastre, F. Bimbot, L.-J. Boë, J. Campbell, D. Reynolds, I. Magrin-Chagnolleau: Person authentication by voice: a need for caution, Proc. Eurospeech (2003) pp. 33–36

36.60 Voice Identification and Acoustic Analysis Subcommittee of the International Association for Identification: Voice comparison standards, J. Forensic Identif. **41**, 373–392 (1991)

36.61 A.E. Rosenberg, S. Parthasarathy, J. Hirschberg, S. Whittaker: Foldering voicemail messages by caller using text independent speaker recognition, Proc. Int. Conf. on Spoken Language Processing (2000)

37. Text-Dependent Speaker Recognition

M. Hébert

Text-dependent speaker recognition characterizes a speaker recognition task, such as verification or identification, in which the set of words (or lexicon) used during the testing phase is a subset of the ones present during the enrollment phase. The restricted lexicon enables very short enrollment (or registration) and testing sessions to deliver an accurate solution but, at the same time, represents scientific and technical challenges. Because of the short enrollment and testing sessions, text-dependent speaker recognition technology is particularly well suited for deployment in large-scale commercial applications. These are the bases for presenting an overview of the state of the art in text-dependent speaker recognition as well as emerging research avenues. In this chapter, we will demonstrate the intrinsic dependence that the lexical content of the password phrase has on the accuracy. Several research results will be presented and analyzed to show key techniques used in text-dependent speaker recognition systems from different sites. Among these, we mention multichannel speaker model synthesis and continuous adaptation of speaker models with threshold tracking. Since text-dependent speaker recognition is the most widely used voice biometric in commercial deployments, several results drawn from realistic deployment scenarios are also included.

37.1 Brief Overview

There exists significant overlap and fundamental differences between text-dependent and text-independent speaker recognition. The underlying technology and algorithms are very often similar. Advances in one field, frequently text-independent speaker recognition because of the NIST evaluations [37.1], can be applied with success in the other field with only minor modifications. The main difference, as pointed out by the nomenclature, is the lexicon allowed by each. Although not restricted to a specific lexicon for enrollment, text-dependent speaker recognition assumes that the lexicon active during the testing is a subset of the enrollment lexicon. This limitation does not exist for text-independent speaker recognition where any word can be uttered during enrollment and testing. The known overlap between the enrollment and testing phase results in very good accuracy with a limited amount of enrollment material (typically less than 8 s of speech). In the case of unknown-text speaker recognition, much more enrollment material is required (typically more than 30 s) to achieve similar accuracy. The theme of *lexical content* of the enrollment and testing sessions is central to text-dependent speaker recognition and will be recurrent during this chapter.

Traditionally, text-independent speaker recognition was associated with speaker recognition on entire conversations. Lately, work from *Sturim* et al. [37.2] and others [37.3] has helped bridge the gap between text-dependent and text-independent speaker recognition by using the most frequent words in conversational speech and applying text-dependent speaker recognition techniques to these. They have shown the benefits of using text-dependent speaker recognition techniques on a text-independent speaker recognition task.

Table 37.1 illustrates the challenges encountered in text-dependent speaker recognition (adapted from [37.4]). It can be seen that the two main sources of degradation in the accuracy are channel and lexical mismatch. Channel mismatch is present in both text-dependent and text-independent speaker recognition, but mismatch in the lexical content of the enrollment and testing sessions is central to text-dependent speaker recognition.

Throughout this chapter, we will try to quantify accuracy based on application data (from trial data collections, comparative studies or live data). We will favor live data because of its richness and relevance. Special care will be taken to reference accuracy on publicly available data sources (some may be available for a fee), but in some other cases an explicit reference is impossible to preserve contractual agreements. Note that a comparative study of off-the-shelf commercial text-dependent speaker verification systems was presented at Odyssey 2006 [37.5].

This chapter is organized as follows. The rest of this section explains at a high-level the main components of a speaker recognition system with an emphasis on particularities of text-dependent speaker recognition. The reader is strongly encouraged, for the sake of completeness, to refer to the other chapters on speaker recognition. Section 37.2 presents the main technical and commercial deployment challenges. Section 37.3 is formed by a collection of selected results to illustrate the challenges of Sect. 37.2. Concluding remarks are found in Sect. 37.4.

Table 37.1 Effect of different mismatch types on the EER for a text-dependent speaker verification task (after [37.4]). The corpus is from a pilot with 120 participants (gender balanced) using a variety of handsets. Signal-to-noise ratio (SNR) mismatch is calculated using the difference between the SNR during enrollment and testing (verification). For the purposes of this table, an absolute value of this difference of more than 10 db was considered mismatched. Channel mismatch is encountered when the enrollment and testing sessions are not on the same channel. Finally, lexical mismatch is introduced when the lexicon used during the testing session is different from the enrollment lexicon. In this case, the password phrase was always a three-digit string. LD0 stands for a lexical match such that the enrolment and testing were performed on the same digit string. In LD2, only two digits are common between the enrollment and testing; in LD4 there is only one common digit. For LD6 (complete lexical mismatch), the enrollment lexicon is disjoint from the testing lexicon. Note that, when considering a given type of mismatch, the conditions are matched for the other types. At EERs around 8%, the 90% confidence interval on the measures is 0.8%

Type of mismatch	Accuracy (EER) (%)
No mismatch	7.02
SNR mismatch	7.47
Channel mismatch	9.76
Lexical mismatch (LD2)	8.23
Lexical mismatch (LD4)	13.4
Complete lexical mismatch (LD6)	36.3

37.1.1 Features

The first text-dependent speaker recognition system descriptions that incorporate the main features of the current state of the art date back to the early 1990s. In [37.6] and [37.7], systems have feature extraction, speaker models and score normalization using a likelihood ratio scheme. Since then, several groups have explored different avenues. The work cited below is not restricted to the text-dependent speaker recognition field, nor is it intended as an exhaustive list. Feature sets usually come in two flavors: MEL [37.8] or LPC (linear predictive coding) [37.6, 9] cepstra. Cepstral mean subtraction and feature warping have proved effective on cellular data [37.10] and are generally accepted as an effective noise robustness technique. The positive role of dynamic features in text-dependent speaker recognition has recently been reported in [37.11]. Finally, a feature mapping approach [37.12] has been proposed as an equivalent to speaker model synthesis [37.13]; this is an effective channel robustness technique.

37.1.2 Acoustic Modeling

Several modeling techniques and their associated scoring schemes have been investigated over the years. By far the most common modeling scheme across

speaker recognition systems is the hidden Markov model (HMM) [37.14]. The unit modeled by the HMM depends heavily on the type of application (Fig. 37.1). In an application where the enrollment and testing lexicon are identical and in the same order (*My voice is my password* as an example), a sentence-level HMM can be used. When the order in which the lexicon appears in the testing phase is not the same as the enrollment order, a word-level unit is used [37.9, 15]. The canonical application of word-level HMMs is present in digit-based speaker recognition dialogs. In these, all digits are collected during the enrollment phase and a random digit sequence is requested during the testing phase. Finally, phone-level HMMs have been proposed to refine the representation of the acoustic space [37.16–18]. The choice of HMMs in the context of text-dependent speaker recognition is motivated by the inclusion of inherent time constraints.

The topology of the HMM also depends on the type of application. In the above, standard left-to-right N-state HMM have been used. More recently, single-state HMMs [also called Gaussian mixture models (GMMs)] have been proposed to model phoneme-level acoustics in the context of text-dependent speaker recognition [37.19] and later applied to text-independent speaker recognition [37.20]. In this case, the temporal information represented by the sequence of phonemes is dictated by an external source (a speech recognition system) and not inscribed in the model's topology. Note that GMMs have been extensively studied, and proved very effective, in the context of text-independent speaker recognition.

In addition to the mainstream HMMs and GMMs, there exists several other modeling methods. Support vector machine (SVM) classifiers have been suggested for speaker recognition by *Schmidt* and *Gish* [37.21] and have become increasingly used in the text-independent speaker recognition field [37.22, 23]. To our knowledge, apart from [37.24, 25], there has been no thorough study of an SVM-based system on a text-dependent speaker recognition task. In this context, the key question is to assess the robustness of an SVM-based system to a restricted lexicon. Dynamic time warping (DTW) algorithms have also been investigated as the basis for text-dependent speaker recognition [37.26, 27]. Finally, neural networks (NNs) modeling methods also form the basis for text-dependent speaker recognition algorithms [37.28, 29]. Since the bulk of the literature and advances on speaker recognition are based on algorithms that are built on top of HMM or GMM, we will focus for the rest of this chapter on those. We believe, however, that the main conclusions and results herein apply largely to the entire field of text-dependent speaker recognition.

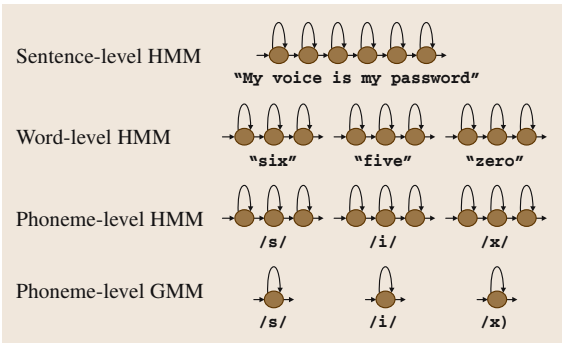


Fig. 37.1 Hidden Markov model (HMM) topologies

37.1.3 Likelihood Ratio Score

As mentioned in a previous section, speaker recognition can be split into speaker identification and verification. In the case of speaker identification, the score is simply the likelihood, template score (in the case of DTW), or posterior probability in the case of an NN. For speaker verification, the standard scoring scheme is based on the competition between two hypothesis [37.30].

- H_0: the test utterance is from the claimed speaker C, modeled by λ;
- H_1: the test utterance is from a speaker other than the claimed speaker C, modeled by $\overline{\lambda}$.

Mathematically, the likelihood ratio $[L(\boldsymbol{X}|\lambda)]$ detector score is expressed as

$$L(\boldsymbol{X}|\lambda) = \log p(\boldsymbol{X}|\lambda) - \log p(\boldsymbol{X}|\overline{\lambda}) \,, \tag{37.1}$$

where $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\mathrm{T}\}$ is the set of feature vectors extracted from the utterance and $p(\boldsymbol{X}|\lambda)$ is the likelihood of observing $\boldsymbol{X}$ given model λ. H_0 is represented by a model λ of the claimed speaker C. As mentioned above, λ can be an HMM or a GMM that has been trained using features extracted from the utterances from the claimed speaker C during the enrollment phase. The representation of H_1 is much more subtle because it should, according to the above, model all potential speakers other than C. This is not tractable in a real system. Two main approaches have been studied to model $\overline{\lambda}$. The first consists of selecting N background or cohort speakers, to model individually $(\lambda_0, \lambda_1, \ldots, \lambda_{N-1})$ and to combine their likelihood score on the test utterance.

The other approach uses speech from a pool of speakers to train a single model, called a general, background or universal background model (UBM). A variant of the UBM, widely used for its channel robustness, is to train a set of models by selecting utterances based on some criteria such as gender, channel type, or microphone type [37.8]. This technique is similar in spirit to the one presented in [37.12]. Note that for the case of text-dependent speaker recognition, it is beneficial to train a UBM with data that lexically match the target application [37.19].

37.1.4 Speaker Model Training

In order to present a conceptual understanding of the text-dependent speaker recognition field, unless otherwise stated, we will assume only two types of underlying modeling: a single GMM for all acoustic events, which is similar to the standard modeling found in text-independent tasks. We will call this the *single-GMM* approach. The other modeling considered is represented as *phoneme-level GMM* on Fig. 37.1. This approach will be called phonetic-class-based verification (PCBV as per [37.19]). This choice is motivated by simplicity, availability of published results, as well as current trends to merge known and text-independent speaker recognition (Sect. 37.1).

For these types of modeling, training of the speaker model is performed using a form of Bayesian adaptation [37.30, 31], which alters the parameters of $\overline{\lambda}$ using the features extracted from the speech collected during the enrollment phase. As will be shown later, this form of training for the speaker models is well suited to allow adaptation coefficients that are different for means, variances, and mixture weights. This, in turn has an impact on the accuracy in the context of text-dependent speaker recognition.

37.1.5 Score Normalization and Fusion

Although the score coming from the likelihood ratio detector (37.1) discriminates genuine speakers from imposters well, it remains fragile. Several score normalization techniques have been proposed to improve robustness. We will discuss a few of those.

The first approach is called the H-norm, which stands for handset normalization, and is aimed at normalizing handset variability [37.32], especially cross-channel variability. A similar technique called the Z-norm has also been investigated in the context of text-dependent speaker recognition with adaptation [37.33]. Using a set ($\simeq 200$) of impostor test utterances with known handset and/or gender labels, a newly trained speaker model is challenged. The scores calculated using (37.1) are fitted using a Gaussian distribution to estimate their mean $[\mu_{\mathrm{H}}(\lambda)]$ and standard deviations $[\sigma_{\mathrm{H}}(\lambda)]$ for each label H. At test time, a handset and/or gender labeler [37.8,32] is used to identify the label H of the testing utterance. The normalized score is then

$$L_{\text{H-norm}}(\boldsymbol{X}|\lambda, H) = \frac{L(\boldsymbol{X}|\lambda) - \mu_{\mathrm{H}}(\lambda)}{\sigma_{\mathrm{H}}(\lambda)} \,. \tag{37.2}$$

This technique is computationally very efficient because $\mu_{\mathrm{H}}(\lambda)$ and $\sigma_{\mathrm{H}}(\lambda)$ are calculated once after the enrollment. It can, however, become inefficient when adaptation of the speaker model (Sect. 37.1.6) occurs because $\mu_{\mathrm{H}}(\lambda)$ and $\sigma_{\mathrm{H}}(\lambda)$ need to be recalculated.

Another score normalization technique widely used is called test normalization or T-norm [37.34]. This approach is applied to text-dependent speaker recognition in [37.35]. It can be viewed as the dual to H-norm in that, instead of challenging the target speaker model with a set imposter test utterances, a set of imposter speaker models (T) are challenged with the target test utterance. Assuming a Gaussian distribution of those scores, $\mu_{\mathrm{T}}(\boldsymbol{X})$ and $\sigma_{\mathrm{T}}(\boldsymbol{X})$ are calculated and applied using

$$L_{\text{T-Norm}}(\boldsymbol{X}|\lambda, T) = \frac{L(\boldsymbol{X}|\lambda) - \mu_{\mathrm{T}}(\boldsymbol{X})}{\sigma_{\mathrm{T}}(\boldsymbol{X})} \,. \tag{37.3}$$

By construction, this technique is computationally very expensive because the target test utterance has to be applied to the entire set of imposter speaker models.

Notwithstanding the computational cost, the H-norm and T-norm developed in the context of text-independent speaker recognition need to be adapted for the text-dependent speaker recognition. These techniques have been shown to be heavily dependent on the lexicon of the set of imposter utterances (H-norm [37.36]) and on the lexicon of the utterances used to train the imposter speaker models (T-norm [37.35]). The issue of lexical dependency or mismatch is not present in a text-independent speaker recognition task, but heavily influences text-dependent speaker recognition system designs [37.4]. We will come back to this question later (Sect. 37.3.2 and Sect. 37.3.4).

Finally, as in the text-independent speaker recognition systems, score fusion is present in text-dependent systems and related literature [37.29]. The goal of fusion is to combine classifiers that are assumed to make uncorrelated errors in order to build a better performing overall system.

37.1.6 Speaker Model Adaptation

Adaptation is the process of extending the enrollment session to the testing sessions. Common wisdom tells us that *the more speech you train with, the better the accuracy will be*. This has to be balanced with requirements from commercial deployments where a very long enrollment sessions is negatively received by end customers. A way to circumvent this is to fold back into the enrollment material any testing utterance that the system has a good confidence of having been spoken by the same person as the original speaker model. Several studies on unknown [37.37, 38] and text-dependent [37.39, 40] speaker recognition tasks have demonstrated the effectiveness of this technique. Speaker model adaptation comes in two flavors. Supervised adaptation, also known as retraining or manual adaptation, implies an external verification method to assess that the current speaker is genuine. That can be achieved using a secret piece of information or another biometric method. The second method is called unsupervised adaptation. In this case, the decision taken by the speaker recognition system (a verification system in this case) is used to decide on the application of adaptation of the speaker model with the current test utterance. Supervised adaptation outperforms its unsupervised counterpart in all studies. A way to understand this fact is to consider that unsupervised adaptation requires a good match between the target speaker model and the testing utterance to adapt the speaker model; hence this new utterances does not bring new variability representing the speaker, the transmission channel, the noise environment, etc. The supervised adaptation scheme, since it is not based on the current utterance, will bring these variabilities to the speaker model in a natural way. Under typical conditions, supervised adaptation can cut, on text-dependent speaker verification tasks, the error rates by a factor of five after 10–20 adaptation iterations.

37.2 Text-Dependent Challenges

The text-dependent speaker recognition field faces several challenges as it strives to become a mainstream biometric technique. We will segregate those into two categories: technological and deployment. The technology challenges are related to the core algorithms. Deployment challenges are faced when bringing the technology into an actual application, accepting live traffic. Several of these challenges will be touched on in subsequent sections where we discuss the current research landscape and a set of selected results (Sect. 37.3). This section is a superset of challenges found in a presentation by *Heck* at the Odyssey 2004 workshop [37.41]. Note that the points raised here can all give rise to new research avenues; some will in fact be discussed in following sections.

37.2.1 Technological Challenges

Limited Data and Constrained Lexicon

As mentioned in Sect. 37.1, text-dependent speaker recognition is characterized by short enrollment and testing session. Current commercial applications use enrollment sessions that typically consist of multiple repetitions (two or three) of the enrollment lexicon. The total speech collected is usually 4–8 s (utterances are longer than that, but silence is usually not taken into account). The testing session consists of a single (or sometimes two) repetitions of a subset of the enrollment lexicon, for a total speech input of 2–3 s. These requirements are driven by usability studies which show that shorter enrollment and testing sessions are best perceived by end customers.

The restricted nature of the lexicon (hence text-dependent speaker recognition), is a byproduct of the short enrollment sessions. To achieve deployable accuracies under the short enrollment and testing constraints, the lexicon has to be restricted tremendously. Table 37.2 lists several examples of enrollment lexicon present is deployed applications. Table 37.3 describes typical testing strategies given the enrollment lexicon. In most cases, the testing lexicon is chosen to match the enrollment lexicon exactly. Note that, for random (and pseudo random) testing schemes, a *2-by-4* approach is sometimes used: in order to reduce the cognitive load, a four-digit string repeated twice is requested from

Table 37.2 Examples of enrolment lexicon

Abreviation	Description
E	Counting from 1 to 9: *one two three* ...
T	10-digit telephone number
S	9-digit account number
N	First and last names
MVIMP	*My voice is my password*

Table 37.3 Examples of testing lexicon. Note that the abbreviations refer to Table 37.2 and each line gives depicts a potential testing lexicon given the enrolment lexicon

Abreviation	Description
E	Counting from 1 to 9: *one two three ...*
R	Random digit sequence *2 6 8 5 2 6 8 5*
pR	Pseudorandom digit sequence from E *2 3 6 7 2 3 6 7*
T	Same 10-digit telephone number as enrolment
	Random digit sequence selected from enrolment lexicon
	Pseudorandom digit sequence selected from enrolment lexicon
S	Similar to T but for a nine-digit account number
N	First and last names
MVIMP	*My voice is my password*

the user. This makes for a longer verification utterance without increasing the cognitive load: a totally random eight-digit string could hardly be remembered by a user. Table 37.4 shows a summary of the accuracy in different scenarios. We reserve discussion of these results for Sect. 37.3.2.

Channel Usage

It is not rare to see end customers in live deployments using a variety of handset types: landline phones, pay phones, cordless phones, cell phones, etc. This raises the issue of their impact on accuracy of channel usage. A cross-channel attempt is defined as a testing session originating from a different channel than the one used during the enrollment session. It is not rare to see the proportion of cross-channel calls reach 25–50% of all genuine calls in certain applications. The effect on the accuracy is very important ranging from doubling the EER [37.4, 42] to quadrupling [37.42] the EER on some commercial deployments. This is a significant area where algorithms must be improved. We will come back to this later.

Aging of Speaker Models

It has been measured in some commercial trials [37.42] and in data collections [37.15] that the accuracy of a text-dependent speaker recognition system degrades slowly over time. In the case of [37.15], the error rate increased by 50% over a period two months. There exists several sources of speaker model aging, the main ones being

Table 37.4 Speaker verification results (EERs) for different lexicon. Refer to Tables 37.2 and 37.3 for explanations of the acronyms. Empty cells represent the fact that pseudorandom strings (pR) do not apply to S since the pseudorandom string is extracted from an E utterance. Italicized results depict conditions that are not strictly text-dependent speaker verification experiments. At EERs of 5–10%, the 90% confidence interval on the measures is 0.3–0.4%

Verify	E	S	R	pR	N
Enroll					
E	6.16%	10.2%	13.2%	10.4%	*36.2%*
S	11.6%	5.05%	14.2%		*39.3%*
R	10.7%	9.43%	11.5%	10.0%	*36.4%*
pR	10.0%		11.3%	8.05%	*35.6%*
N	*38.9%*	*39.7%*	*39.3%*	*39.1%*	10.6%

natural aging, channel usage, and behavioral changes. Natural aging is related to the physiological changes that occur to the phonatory apparatus over long periods of time. Channel usage changes over time can cause the speaker model to become outdated with respect to the current channel usage. Finally, behavioral changes occur when users get more exposure to the voice interface and thus alter the way in which they interact with it. As an example, first-time users of a speech application (usually the enrollment session) tend to cooperate with the system by speaking slowly under friendly conditions. As these users get more exposure to the application, they will alter the way that they interact with it and use it in adverse conditions (different channels, for example). All of these factors affect the speaker models and scoring, and thus are reflected in the accuracy. The common way to mitigate this effect is to use speaker model adaptation (Sect. 37.1.6 and Sect. 37.3.5).

37.2.2 Commercial Deployment Challenges

Dialog Design

One of the main pitfalls in deploying a speech-based security layer using text-dependent speaker recognition is poor dialog design choices. Unfortunately, these decisions are made very early in the life cycle of an application and have a great impact on the entire life of the application. Examples [37.41] are

1. small amount of speech collected during enrollment and/or verification
2. speech recognition difficulty of the claim of identity (such as a first and last names in a long list)

3. poor prompting and error recovery
4. lexicon mismatch between enrollment and verification

One of the challenges in deploying a system is certainly protection against recordings since the lexicon is very restricted. As an example, in the case where the enrollment and verification lexicon is *My voice is my password* or a telephone number, once a fraudster has gained access to a recording from the genuine speaker, the probability that they can gain access has greatly increased. This can be addressed by several techniques (one can also think about combining them). The first technique consists of explicitly asking for a randomized subset of the lexicon. This does not lengthen the enrollment session and is best carried out if the enrollment lexicon consists of digits. The second is to perform the verification process across the entire dialog even if the lexical mismatch will be high (Sect. 37.3.2 and Sect. 37.3.6), while maintaining a short enrollment session. A third technique is to keep a database of *trusted* telephone numbers for each user (home, mobile, and work) and to use this external source of knowledge to improve security and ease of use [37.43]. Finally, a challenge by a secret *knowledge* question drawn from a set of questions can also be considered. These usually require extra steps during the enrollment session. It is illusory to think that a perfect system (no errors) can be designed, the goal is simply to raise the bar of

1. the amount of information needed and
2. the sophistication required by a fraudster to gain access.

There are two other considerations that come into play in the design of an application. The first is related to the choice of the token for the identity claim in the case of speaker verification. The identity claim can be combined with the verification processing in systems that have both speaker and speech recognition. In this case, an account number or a name can be used. As can be seen from Table 37.4, verification using text (first and last names) is challenging, mainly due to the short length of speech. For a very large-scale deployment, recognition can also be very challenging. *Heck* and *Genoud* have suggested combining verification and recognition scores to re-sort the N-best list output from the recognizer and achieve significant recognition accuracy gains [37.44]. Other means of claiming an identity over the telephone include caller identification (ID) and keypad input. In these cases, the verification utterance can be anything, including a lexicon common to all users. The second consideration is the flexibility that the enrollment lexicon provides to dynamically select a subset of the lexicon with which to challenge the user in order to protect against recordings (see above). This is the main reason why digit strings (telephone and account number, for example) are appealing for a relatively short enrollment session. A good speaker model can be built to deliver good accuracy even with a random subset of the enrollment lexicon as the testing lexicon (Sect. 37.3.2).

Cost of Deployment

The cost of deploying a speaker recognition system into production is also a challenge. Aside from dialog design and providing the system with a central processing unit (CPU), storage, and bandwidth, setting the operating point (the security level or the target false-acceptance rate) has a major impact on cost. As can be seen from the discussion above, there are a wide variety of dialogs that can be implemented, and all of these require their own set of thresholds depending on the level of security required. This is a very complex task that is usually solved by collecting a large number of utterances and hiring professional services from the vendor to recommend those thresholds. This can be very costly for the application developer. Recently, there has been an effort to build off-the-shelf security settings into products [37.36]. This technique does not require any data and is accurate enough for small- to medium-scale systems or initial security settings for a trial. Most application developers, however, want to have a more-accurate picture of the accuracy of their security layer and want a measurement on actual data of the standard false accept (FA), false reject (FR), and reprompt rates (RR, the proportion of genuine speakers that are reprompted after the first utterance). To this end a data collection is set up. The most expensive portion of data collection is to gather enough impostor attempts to set the decision threshold to achieve the desired FA rate with a high level of confidence. Collecting genuine speaker attempts is fairly inexpensive by comparison. An algorithm aimed at setting the FA rate without specifically collecting impostor attempts has been presented [37.45]. See Sect. 37.3.7 for more details.

Forward Compatibility

Another challenge from a deployment perspective, but that has ramifications into the technology side, is forward compatibility. The main point here is that the database of enrollee (those that have an existing speaker model) should be forward compatible to revision of: (a) the

application, and (b) the software and its underlying algorithms. Indeed, an application that has been released using a security layer based on a first name and last name lexicon is confined to using this lexicon. This is very restrictive. Also, in commercial systems, the enrollment utterances are not typically saved: the speaker model is the unit saved. This speaker model is a parameterized version of the enrollment utterances. The first step that goes into this parameterization is the execution of the front-end feature extractor (Sect. 37.1.1). The definition of these features is an integral part of the speaker model and any change to this will have a negative impact on accuracy. This also restricts what research can contribute to an existing application.

37.3 Selected Results

In this section, we will present several results that either support claims and assertions made earlier or illustrate current challenges in text-dependent speaker recognition. It is our belief that most if not all of these represent potential areas for future advances.

37.3.1 Feature Extraction

Some of the results presented below are extracted from studies done on text-independent speaker recognition tasks. We believe that the algorithms presented should also be beneficial to text-dependent tasks, and thus could constitute the basis for future work.

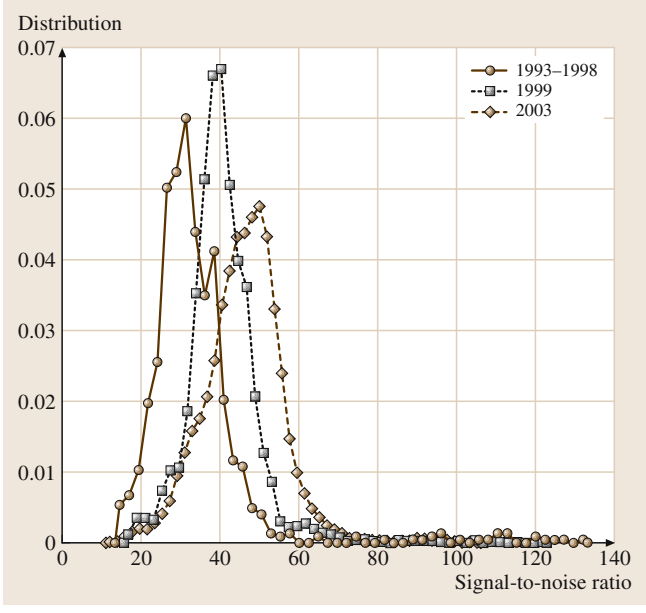


Fig. 37.2 Signal-to-noise ratio distribution from cellular waveforms for three different periods. The data are from a mix of in-service data, pilot data, and data collection

Impact of Codecs on Accuracy

The increasing penetration of cellular phones in society has motivated researchers to investigate the impact of different codecs on speaker recognition accuracy. In 1999, a study of the impact of different codecs was presented [37.46]. Speech from an established corpora was passed through different codecs (GSM, G.729 and G723.1), and resynthesized. The main conclusion of this exercise was that the accuracy drops as the bit rate is reduced. In that study, speaker recognition from the codec parameters themselves was also presented.

Figure 37.2 presents the distribution of the signal-to-noise ratio (SNR) from different internal corpora (trials and data collections) for cellular data only. We have organized them by time periods. A complete specification of those corpora is not available (codecs used, environmental noise conditions, analog versus digital usage, etc.). Nevertheless, it is obvious that speech from cellular phones is cleaner in the 2003 corpora than ever before. This is likely due to more-sophisticated codecs and better digital coverage. It would be interesting to see the effect on speaker recognition and channel identification of recent codecs like CDMA (code division multiple access) in a study similar to [37.46]. This is particularly important for commercial deployments of (text-dependent) speaker recognition, which are faced with the most up-to-date wireless technologies.

Feature Mapping

Feature mapping was introduced by *Reynolds* [37.12] to improve channel robustness on a text-independent speaker recognition task. Figure 37.3a describes the offline training procedure for the background models. The root GMM is usually trained on a collection of utterances from several speakers and channels using k-means and EM (expectation maximization) algorithms. MAP (maximum a posteriori) adaptation [37.31] is used to adapt the root GMM with utterances coming from single channels to produce GMMs for each channel. Because of

the MAP adaptation structure, there exists a one-to-one correspondence between the Gaussians from the root and channel GMMs, and transforms between Gaussians of these GMMs can be calculated [37.12]. The transforms from the channel GMMs Gaussians to the root GMM Gaussians can be used to map features from the those channels onto the root GMM. The online procedure is represented on Fig. 37.3b. For an incoming utterance, the channel is first selected by picking the most likely over the entire utterance based on $\log p(\boldsymbol{X}|\overline{\lambda})$ from (37.1). The features are then mapped from the identified channel onto the root GMM. At this point, during training of the speaker model, mapped features are used to adapt the root GMM. Conversely, during testing, the mapped features are used to score the root and speaker model GMMs to perform likelihood-ratio scoring (Sect. 37.1.3).

Feature mapping has proved its effectiveness for channel robustness (see [37.12] for more details). It is of interest for text-dependent speaker recognition because it is intimately related to speaker model synthesis (SMS) [37.13], which has demonstrated its effectiveness for such tasks [37.40]. To our knowledge, feature mapping has never been implemented and tested on a text-dependent speaker recognition task.

Speaker and Speech Recognition Front Ends

The most common feature extraction algorithms for speech recognition are mel-filter cepstral coefficients (MFCCs) and linear predictive cepstral coefficients (LPCCs). These algorithms have been developed with the objective of classifying phonemes or words (lexicon) in a speaker-independent fashion. The most common feature extraction algorithms for speaker recognition are, surprisingly, MFCC or LPCC. This is surprising because of the fact that speaker recognition objective is the classification of speakers, with no particular emphasis on lexical content. A likely, but still to be proven, explanation for this apparent dichotomy is that MFCC and LPCC are very effective at representing a speech signal in general. We believe that other approaches are worth investigating.

Several studies have tried to change the speaker recognition paradigm for feature extraction (see [37.47, 48], to name a few). In [37.47], a neural net with five layers is discriminatively trained to maximize speaker discrimination. Then the last two layers are discarded and the resulting final layer constitutes the feature extractor. Authors report a 28% relative improvement over MFCCs in a text-independent speaker recognition task.

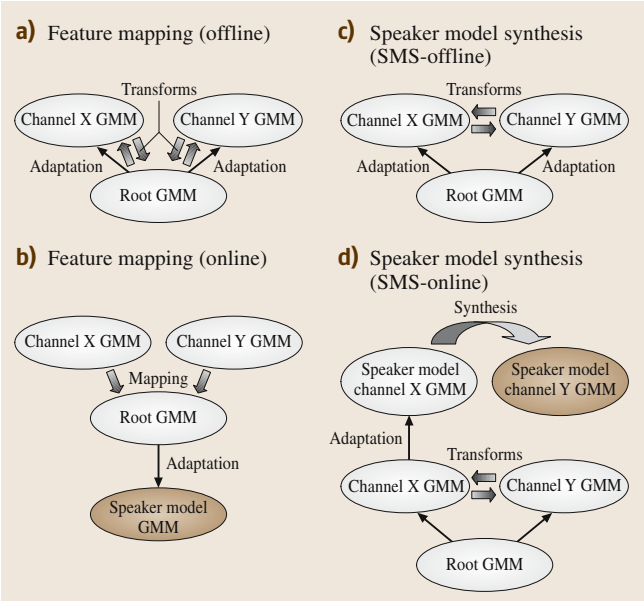


Fig. 37.3a–d Feature mapping and speaker model synthesis (SMS). GMMs with oblique lines were constructed using synthesized data

Although developed with channel robustness in mind, we believe that this technique holds a lot of potential. In [37.48], wavelet packet transforms are used to analyze the speech time series instead of the standard Fourier analysis. Authors report a 15–27% error rate reduction on an text-independent speaker recognition task. Despite the improvements reported, these algorithms have not reached mainstream adoption to replace MFCCs or LPCCs.

37.3.2 Accuracy Dependence on Lexicon

As mentioned in the Chap. 36, the theme of the lexical content of the password phrase is central in text-dependent speaker recognition. A study by *Kato* and *Shimizu* [37.15] has demonstrated the importance of preserving the sequence of digits to improve accuracy. The authors report a relative improvement of more than 50% when the digit sequence in the testing phase preserves the order found during enrollment.

Table 37.4 presents a similar trend as well as additional conditions. The data for these experiments was collected in September of 2003 from 142 unique speakers (70 males and 72 females). Each caller was requested to complete at least four calls from a variety of handsets (landline, mobile, etc.) in realistic noise conditions. In each call, participants were requested to read a sheet

with three repetitions of the phrases E, S, R, pR, and N (refer to Tables 37.2 and 37.3 for explanations of the acronyms). There were only eight unique S digit strings in the database in order to use round-robin imposter attempts, and a given speaker was assigned only one S string. The interesting fact about this data set is that we can perform controlled experiments: for every call, we can substitute E for S and vice versa, or any other types of utterances. This allows the experimental conditions to preserve:

1. callers
2. calls (and thus noise and channel conditions) and vary lexical content only

The experiments in Table 37.4 are for speaker verification and the results presented are the equal error rates (EERs). All results are on 20 k genuine speakers attempts and 20 k imposter attempts. The enrollment session consists of three repetitions of the enrollment token while the testing sessions has two repetitions of the testing token. Let us define and use the following notation to describe an experiment's lexical content for the enrollment and verification: eXXX_vYY, which defines the enrollment as three repetitions of X, and the testing attempts as two repetitions of Y. For example, the EER for eEEE_vRR is 13.2%.

The main conclusion of *Kato* and *Shimizu* [37.15] are echoed in Table 37.4: sequence-preserving digit strings improves accuracy. Compare the EERs for eEEE_vRR with eEEE_vpRpR. Also, eEEE_vEE, eSSS_vSS, epRpRpR_vpRpR, and eNNN_vNN all perform better than eRRR_vRR. This illustrates the capture by the speaker model of coarticulation: E and R utterances have exactly the lexicon (1 to 9) but in a different order. Note that the accuracy of the first and last names is significantly worse than E or S on the diagonal of Table 37.4. This is due to the average length of the password phrase: an E utterance has on average 3.97 s of speech while an N utterance has only 0.98. Finally, we have included cross-lexicon results, which are more relevant to text-independent speaker recognition (for example eEEE_vNN). This illustrates the fact that, with very short enrollment and verification sessions, lexically mismatched attempts impact accuracy significantly. In [37.4], the effect of lexical mismatch is compared with the effect of SNR mismatch and channel mismatch. It is reported that a moderate lexical mismatch can degrade the accuracy more than SNR and is comparable to channel mismatch (Table 37.1). Finally, *Heck* [37.41] noted that 'advances in robustness to linguistic mismatches will form a very fruitful bridge between text-independent and dependent tasks.' We share this view and add that solving this problem would open avenues to perform accurate and non-intrusive protection against recordings by verifying the identity of a caller across an entire call even with a very short enrollment session. We will explore this more in Sect. 37.3.6.

37.3.3 Background Model Design

The design of background models is crucial to the resulting accuracy of a speaker recognition system. The effect of the lexicon can also be seen in this context. As an example, in a text-dependent speaker recognition task based on *My voice is my password* (MVIMP) as the password phrase, adapting a standard background model with utterances of the exact target lexicon can have a significant positive impact. Without going into the details of the data set, the EER drops from 16.3% to 11.8% when 5 k utterance of MVIMP were used to adapted the background model. This is consistent with one of the results from [37.19]. In [37.49], an algorithm for the selection of background speakers for a target user is presented as well as results on a text-dependent task. The algorithm is based on similarity between two users' enrollment sessions. Lexical content was not the focus of that study, but it would be interesting to see if the lexical content of each enrollment sessions had an influence on the selection of competitive background speakers, i. e., whether similar speakers have significant lexical overlap.

From the point of view of commercial deployments, the use of specialized background models for each password phrase, or on a per-target user basis, is unrealistic. New languages also require investments to develop language-specific background models. The technique in [37.50] does not require offline training of the background model. The enrollment utterances are used to train the 25-state HMM speaker model and a lower-complexity background model. The reasoning behind this is that the reduced complexity model will smear the speaker characteristics that are captured by the higher-complexity model (speaker model). Unfortunately, this technique has never been compared to a state-of-the-art speaker recognition system.

37.3.4 T-Norm in the Context of Text-Dependent Speaker Recognition

As mentioned in Sect. 37.1.5, the T-norm is sensitive to the lexicon of the utterances used to train the imposter speaker models composing the cohort [37.45]. In that study, the data used is a different organization of the data set described in Sect. 37.3.2 that allows a separate set of speakers to form the cohort needed by the T-norm. The notation introduced in Sect. 37.3.2 can also be adapted to describe the lexicon used for the cohort: eXXX_vYY_cZZZ describes an experiment for which the speaker models in the cohort are enrolled with three repetitions of Z. The baseline system used for the experiments in that study is described in *Teunen* et al. [37.13]. It uses gender- and handset-dependent background models with speaker model synthesis (SMS). The cohort speaker models are also tagged with gender and handset; the cohorts are constructed on a per-gender and per-handset basis. During an experiment, the selection of the cohort can be made after the enrollment session based on the detected handset and gender from the enrollment session. It can also be made at test time using the handset and gender detected from the testing utterance. We denote the set of cohorts selected at testing by C_t. In the results below, we consider only the experiments eEEE_vEE or eSSS_vSS with lexically rich (cSSS) or lexically poor cohorts (cEEE). A note on *lexically rich and poor* is in order: the richness comes from the variety of contexts in which each digit is found. This lexical richness in the cohort builds robustness with respect to the variety of digits strings that can be encountered in testing.

Table 37.5 shows the accuracy using test-time cohort selection C_t in a speaker verification experiment. It is interesting to note that the use of a lexically poor cohort (cEEE) in the context of an eSSS_vSS experiment significantly degrades accuracy. In all other cases in Table 37.5, the T-norm improves the accuracy. A smoothing scheme was introduced to increase robustness to the lexical poorness of the cEEE cohort. It is suggested that this smoothing scheme increases the robustness to lexical mismatch for the T-norm. The smoothing scheme is based on the structure of (37.1), which can be rewritten in a form similar to (37.3) using $\mu(\boldsymbol{X}) = \log p(\boldsymbol{X}|\overline{\lambda})$ and $\sigma(\boldsymbol{X}) = 1$. The smoothing is then an interpolation of the normalizing statistics between standard T-norm [$\mu_{\mathrm{T}}(\boldsymbol{X})$ and $\sigma_{\mathrm{T}}(\boldsymbol{X})$] and background model normalization [$\log p(\boldsymbol{X}|\overline{\lambda})$ and 1]. Figure 37.4 shows DET (detecttion error trade-off) curves for the eSSS_vSS experiment with different cohorts. It is shown

Table 37.5 The FR rates at FA = 1% for various configurations [37.35]. Based on the lower number of trials (the impostor in our case), the 90% confidence interval on the measures is 0.6%. (© 2005 IEEE)

Experimental set-up	Baseline (no T-norm)	T-norm C^t cEEE	T-norm C^t cSSS
eEEE_vEE	17.10%	14.96%	14.74%
eSSS_vSS	14.44%	16.39%	10.42%

that the T-norm with a cEEE cohort degrades the accuracy compared to the baseline (no T-norm) as mentioned above. Smoothed T-norm achieves the best accuracy irrespective of the cohort's lexical richness (a 28% relative improvement of FR at fixed FA).

37.3.5 Adaptation of Speaker Models

Online adaptation of speaker models [37.39,40] is a central component of any successful speaker recognition application, especially text-dependent tasks because of the short enrollment sessions. The results presented in this section all follow the same protocol [37.40]. Unless otherwise stated, the data comes from a Japanese digit data collection. There were 40 speakers (gender balanced) making at least six calls: half from landlines

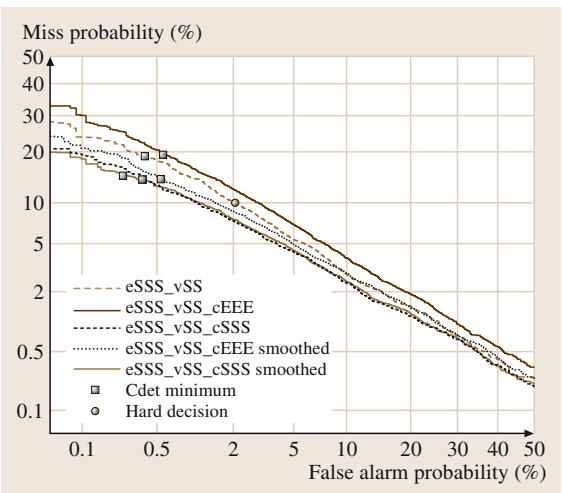


Fig. 37.4 DET curve showing the T-norm and its smoothed variant in the case of eSSS_vSS with the cohort selected at testing time (C^t). The interested reader should refer to the source paper for additional details. For clarity, the order in the legend is the same as the order of the curves at false alarm probability equal to 0.1% (after [37.35], ©2005 IEEE)

and have from cellular phones. The data was heavily recycled to increase the number of attempts by enrolling several speaker models for a given speaker and varying the enrollment lexicon (130–150 on average). For any given speaker model, the data was divided into three disjoint sets: an enrollment set to build the speaker model, an adaptation set, and a test set. The adaptation set was composed of one imposter attempt for every eight genuine attempts (randomly distributed). The experiments were designed as follows. First all of the speaker models were trained and the accuracy was measured right after the enrollment using the test set. Then, one adaptation utterance was presented to each of the speaker models. At this point a decision to adapt or not was made (see below). After this first iteration of adaptation, the accuracy was measured using the test set (without the possibility of adaptation on the testing data). The adaptation and testing steps were repeated for each adaptation iterations in the adaptation set. This protocol was designed to control with great precision all the factors related to the adaptation process: the accuracy was measured after each adaptation iteration using the same test set and they are therefore directly comparable.

Two different types of adaptation experiments can be designed based on how the decision to update the speaker models is made: supervised and unsupervised [37.39]. Both types give insight into the adaptation process and its effectiveness, and both have potential applicability in commercial deployments. Supervised adaptation experiments use the truth about the source of the adaptation utterances: an utterance is used for updating a speaker model only when it is from the target speaker. This allows the update process of the speaker models to be optimal for two reasons. The first is that there is no possibility of corruption of a speaker model by using utterances from an imposter. The second comes from the fact that all adaptation utterances from the target speaker are used to update speaker model. This allows more data to update the speaker model, but more importantly it allows poorly scoring utterances to update the speaker model. Because these utterances score poorly, they have the most impact on accuracy because they bring new and unseen information (noise conditions, channel types, etc.) into the speaker model. This has a significant impact on the cross-channel accuracy, as we will show below. Supervised adaptation can find its applicability in commercial deployments in a scenario where two-factor authentication is used, where one of the factors is acoustic speaker recognition. As an example, in a dialog where acoustic speaker recognition and authentication using a secret challenge question are used, supervised adaptation can be used if the answer to the secret question is correct.

In unsupervised adaptation, there is no certainty about the source of the adaptation utterance and usually the score on the adaptation utterance using the non-updated speaker model is used to make the decision to adapt or not [37.33, 36, 39, 40]. A disadvantage of this approach is the possibility that the speaker models may become adapted on imposter utterances that score high. This approach also reduces the number of utterances that are used to update the speaker model. More importantly, it reduces the amount of new and unseen information that is used for adaptation because this new and unseen information will likely score low on the existing speaker model and thus not be selected for adaptation.

Variable Rate Smoothing

Variable rate smoothing (VRS) was introduced in [37.30] for text-independent speaker recognition. The main idea is to allow means, variances, and mixture weights to be adapted at different rates. It is well known that the first moment of a distribution takes fewer samples to estimate than the second moment. This should be reflected in the update equations for speaker model adaptation by allowing the smoothing coefficient to be different for means, variances, and mixture weights. The authors reported little or no gains on their task. However, VRS should be useful for text-dependent speaker recognition tasks due to the short enrollment sessions. Please refer to [37.30] for the details. In [37.51], VRS was

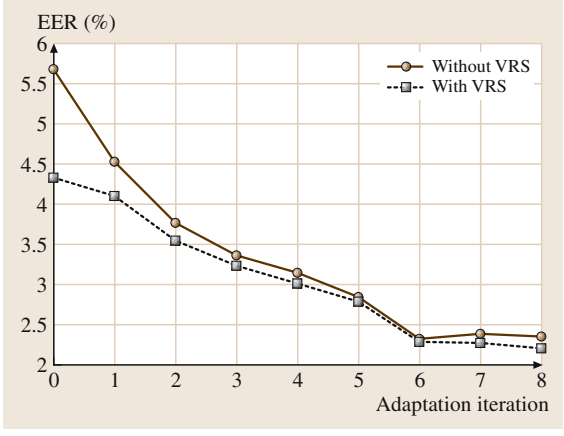


Fig. 37.5 The effect of unsupervised adaptation on the EER (percentage) with and without variable rate smoothing. Adaptation iteration 0 is the enrollment session. Based on the lower number of trials (genuine in our case), the 90% confidence interval on the measures is 0.3%. (After [37.51])

applied to text-dependent speaker recognition; Fig. 37.5 was adapted from that publication. It can be seen that, after the enrollment (iteration 0 on the graph), VRS is most effective because so little data has been used to train the speaker model: smoothing of the variances and mixture weights is not as aggressive as for means because the system does not have adequate estimates. As adaptation occurs, the two curves (with and without VRS) converge: at this point the estimates for the first and second moment in the distributions are accurate, the number of samples is high, and the presence of different smoothing coefficients becomes irrelevant.

Random Digit Strings

We now illustrate the effect of speaker model adaptation on contextual lexical mismatch for a digit-based speaker verification task. The experimental set-up is from a different organization of the data from Sect. 37.3.2 to follow the aforementioned adaptation protocol. Figure 37.6 illustrates the results. The testing is performed on a pseudorandom digit string (see Table 37.3 for details). Enrollment is either performed on a fixed digit string (eEEE) or on a series on pseudorandom digit strings (epRpRpR). Before adaptation occurs, the accuracy of epRpRpR is better than eEEE because the enrollment lexical conditions are matched to testing. However, as adaptation occurs and more pseudorandom utterances are added to the eEEE speaker model, the two curves converge. This shows the power of adaptation to reduce lexical mismatch and to alter the *enrollment* lexicon: in this context, the concept of *enrollment* lexicon becomes fuzzy as adaptation broadens the lexicon that was used to train the speaker model.

Speaker Model Synthesis and Cross-Channel Attempts

Speaker model synthesis (SMS) [37.13] is an extension of handset-dependent background modeling [37.8]. As mentioned before, SMS and feature mapping are dual to each another. Figure 37.3c presents the offline component of SMS. It is very similar to the offline component of feature mapping except that the transforms for means, variances, and mixture weights are derived to transform sufficient statistics from one channel GMM to another rather than from a channel GMM to the root GMM. During online operation, in enrollment, a set of utterances are tagged as a whole to a specific channel (the likeliest channel GMM – the enrollment channel). Then speaker model training (Sect. 37.1.4) uses adaptation with variable rate smoothing [37.30,51] of the enrollment channel GMM. The transforms that have been derived offline are then used at test time to synthesize the enrolled channel GMM across *all* supported channels (Fig. 37.3d). The test utterance is tagged using the same process as enrollment by picking the likeliest channel GMM (the testing channel). The speaker model GMM for the testing channel and the testing channel GMM are then used in the likelihood ratio scoring scheme described in Sect. 37.1.3 and (37.1).

The power of speaker model adaptation (Sect. 37.1.6) when combined with SMS is its ability to synthesize

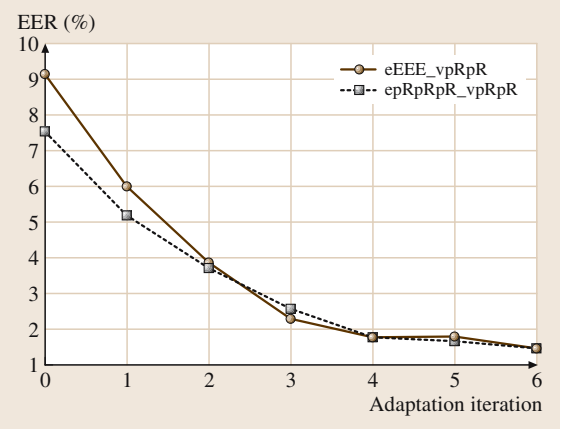


Fig. 37.6 The effect of unsupervised adaptation on reducing the contextual lexical mismatch as depicted by a reduction of the EER. Adaptation iteration 0 is the enrollment session. Based on the lower number of trials (genuine in our case), the 90% confidence interval on the measures is 0.3%

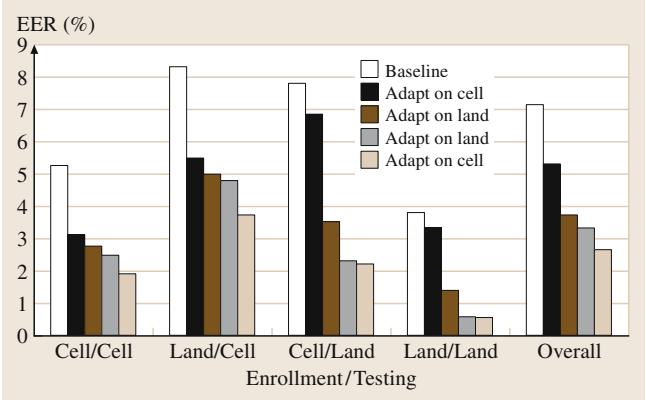


Fig. 37.7 The effect of speaker model adaptation and SMS on the cross-channel accuracy (EER). The interested reader should refer to this paper for additional details. The baseline is the enrollment session. Based on the lower number of trials (genuine in our case), the 90% confidence interval on the measures is 0.6%. (After [37.40])

sufficient statistics across all supported channels. For example, assume that a speaker is enrolled on channel X and a test utterance is tagged as belonging to channel Y. Then, if the test utterance is to be used for adaptation of the speaker model, the sufficient statistics from that utterance is gathered. The transform from Y → X is used to synthesize sufficient statistics from channel Y to channel X before adaptation of the speaker model (on channel X) occurs. Concretely, this would allow adaptation utterances from channel Y to improve the accuracy on all other channels.

Figure 37.7 illustrates the effect of speaker model adaptation with SMS. Results are grouped in *enrollment/testing* conditions: within a group, the enrollment and testing channels are fixed, the only variable is the adaptation material. For each group, the first bar is the accuracy after enrollment. The second bar is the accuracy after one iteration of adaptation on cellular data. The third bar shows the accuracy after the iteration of adaptation on cellular data followed by an iteration on a landline data, and so on. Note that these results are for supervised adaptation and thus an iteration of adaptation on a given speaker model necessarily means an actual adaptation of the speaker model. There are two interesting facts about this figure. The first important feature is that the biggest relative gain in accuracy is when the channel for the adaptation data is matched with the previously unseen testing utterance channel (see the relative improvements between the first and second bars in the *cell/cell* and *land/cell* or between the second and third bars in the *cell/land* and *land/land* results). This is expected since the new data is matched to the (previously unseen) channel of the testing utterance. The other important feature illustrates that the SMS (resynthesis of sufficient statistics) has the ability to improve accuracy even when adaptation has been performed on a different channel than the testing utterance. As an example, in the first block of Fig. 37.7, there is an improvement in accuracy between the second and third bars. The difference between the second and third bars is an extra adaptation iteration on *land* (landline data), but note that the testing is performed on *cell*. This proves that the sufficient statistics accumulated on the *land* channel have been properly resynthesized into the *cell* channel.

Setting and Tracking the Operating Point

Commercial deployments are very much concerned with the overall accuracy of a system but also the operating point, which is usually a specific false-acceptance rate. As mentioned earlier, setting the operating point for a very secure large-scale deployed system is a costly exercise, but for internal trials and low-security solutions, an approximate of the ideal operating point is acceptable. In [37.36] and later in [37.52], a simple algorithm to achieve this has been presented: frame-count-dependent thresholding (FCDT). The idea is simple: parameterize the threshold to achieve a target FA rate as a function of

1. the length of the password phrase
2. the maturity of the speaker model (how well it is trained)

At test time, depending on the desired FA rate, an offset is applied to the score (37.1). Note that the applied offset is speaker dependent because it depends on the length of the password phrase and the maturity of the speaker model.

This parameterization has been done on a large Japanese corpora. The evaluation was conducted on 12 test sets from different languages composed of data collection, trial data and in-service data [37.36]. The operating point for the system was set up at a target FA rate of 0.525% using the above algorithm. The average of the actual FA rates measured was 0.855% with a variance of 0.671%; this new algorithm outperformed previous algorithms [37.33].

In the context of adaptation of the speaker model, the problem of setting an operating point is transformed into a problem of maintaining a constant operating point for all speakers at all times [37.37]. Note that a similar problem arises in the estimation of confidence in speech recognition when adaptation of the acoustic

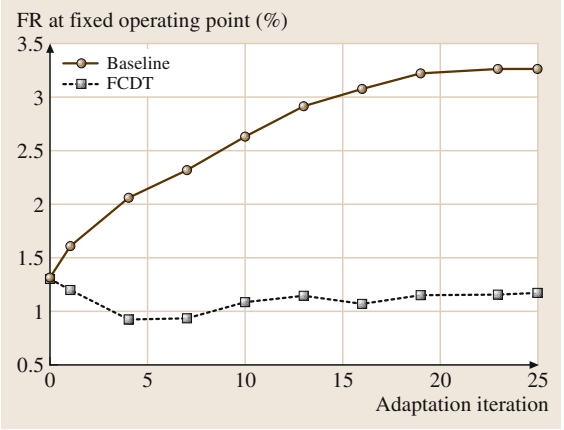


Fig. 37.8 The effect of speaker model adaptation on the FA rate with and without frame-count-dependent thresholding (FCDT). Adaptation iteration 0 is the enrollment session. The 90% confidence interval on the measures is 0.3%. (After [37.36])

models is performed [37.53]. FCDT, as well as other algorithms [37.33], can perform this task. Figure 37.8 presents the false-acceptance rate at a fixed operating point as a function of unsupervised adaptation iterations for an English digits task. After enrollment, both systems are calibrated to operate at FA = 1.3%. Then adaptation is performed. We can very easily see that the scores of the imposter attempts drift towards higher values, and hence the FA rate does not stay constant: the FA rate has doubled after 10 iterations. For commercial deployments, this problem is a crucial one: adaptation of the speaker models is very effective to increase the overall accuracy, but it must not be at the expense of the stability of the operating point. FCDT accomplishes this task: the FA rate stays roughly constant across the adaptation cycles. This leads us to think that FCDT is an effective algorithm to normalize scores against the levels of maturity of the speaker models.

Note that the imposter score drift towards higher values during speaker model adaptation in text-dependent tasks is the opposite behavior from the case of text-independent tasks [37.38, Fig. 3]. This supports the assertion that the existence of a restricted lexicon for text-dependent models has a significant impact on the behavior of speaker recognition systems: both text-dependent [37.36] and text-independent [37.38] systems being GMM-based. During the enrollment and adaptation sessions, several characteristics of the speech signal are captured in the speaker model: the speaker's intrinsic voice characteristics, the acoustic conditions (channels and noise), and the lexicon. In text-dependent speaker recognition, because of the restricted lexicon, the speaker model becomes a lexicon recognizer (the *mini-recognizer* effect). This effect increases the imposter scores because they use the target lexicon.

The FCDT algorithm can be implemented at the phone level in order to account for cases where the enrollment session (and/or speaker model adaptation) does not have a consistent lexicon. In [37.36], all experiments were carried out with enrollment and testing sessions that used exactly the same lexicon for a given user; this might seem restrictive. In the case of phone-level FCDT, the FCDT algorithm would be normalizing maturities of phone-level speaker models.

In the literature on T-norm (for text-dependent or text-independent systems; see Sect. 37.3.4), the speaker models composing the cohorts were all trained with roughly the same amount of speech. In light of the aforementioned results, this choice has the virtue of normalizing against different maturities of speaker models. We believe that the FCDT algorithm can also be used in the context of the T-norm to achieve this normalization.

37.3.6 Protection Against Recordings

As mentioned, protection against recordings is important for text-dependent speaker recognition systems. If the system is *purely* text dependent (that is the enrollment and testing utterances have the same lexicon sequence), once a fraudster has gained access to a recording, it can become relatively easy to break into an account [37.42]. This, however, must be put in perspective. A high-quality recording of the target speaker's voice is required as well as digital equipment to perform the playback. Furthermore, for any type of biometric, once a *recording* and playback mechanism are available the system becomes vulnerable. The advantage that voice authentication has over any other biometrics is that it is natural to prompt for a different sequence of the enrollment sequence: this is impossible for iris scans, fingerprints, etc. Finally, any nonbiometric security layer can be broken into almost 100% of the time once a *recording* of the secure token is available (for example, somebody who steals a badge can easily access restricted areas).

Several studies that assess the vulnerability of speaker recognition systems to altered imposter voices have been published. The general paradigm is that a fraudster gains access to recordings of a target user. Then using different technique the imposter's voice is altered to sound like the target speaker for any password phrase. An extreme case is a well-trained text-to-speech (TTS) system. This scenario is unrealistic because the amount of training material required for a good-quality TTS voice is on the order of hours of high-quality, phonetically balanced recorded speech. Studies along these lines, but using a smaller amount of data, can be found in [37.54, 55]. Even if these studies report the relative weakness of GMM-based speaker recognition systems, these techniques require sophisticated signal processing software and expertise to perform experimentation, along with high-quality recordings. A more-recent study [37.56] has also demonstrated the effect of speech transformation on imposter acceptance. This technique, again, requires technical expertise and complete knowledge of the speaker recognition system (feature extraction, modeling method, UBM, target speaker model, and algorithms). This is clearly beyond the grasp of fraudsters because implementations of security systems are usually kept secret, as are the internals algorithms of commercial speaker recognition systems.

Speaker Recognition Across Entire Calls

Protection against recordings can be improved by performing speaker recognition (in this case verification) across entire calls. The results presented here illustrate a technique to implement accurate speaker recognition across entire calls with a short enrollment session (joint unpublished work with Nikki Mirghafori). It relies heavily on speaker model adaptation (Sect. 37.1.6) and PCBV (Sect. 37.1.4). The verification layer is designed around a password phrase such as an account number. The enrollment session is made up of three repetitions of the password phrase only, while the testing sessions are composed of one repetition of the password phrase followed by non-password phrases. This is to simulate a dialog with a speech application after initial authentication has been performed. Adaptation is used to learn new lexical items that were not seen during enrollment and thus improve the accuracy when non-password phrases are used. The choice for this set-up is motivated by several factors. This represents a possible *upgrade* for currently deployed password-based verification application. It is also seamless to the end user and does not require re-enrollment: the non-password phrases are learnt using speaker model adaptation during the verification calls. Finally it is believed that this technique represents a very compelling solution for protection against recordings.

Note that this type of experiment is at the boundary between text-dependent and text-independent speaker recognition because the testing session is cross-lexicon for certain components. It is hard to categorize this type of experimental set-up because the enrollment session is very short and lexically constrained compared to its text-independent counterpart. Also, the fact that some testing is made cross-lexicon means that it does not clearly belong to the text-dependent speaker recognition field.

In order to benchmark this scenario, Japanese and Canadian French test sets were set up with eight-digit strings (account number) as the *password phrase*. The initial enrollment used three repetitions of the password phrase. We benchmark accuracy on the password and on non-password phrases. In these experiments, the non-password phrases were composed of general text such as first/last names, dates, and addresses. For adaptation, we used the same protocol as in Sect. 37.3.5 with a held-out set composed of non-password phrases (supervised adaptation). Section 37.3.5 has already demonstrated the effectiveness of adaptation on password phrases; these results show the impact, on both password and non-password phrases, of adapting on non-password phrases. Figure 37.9 presents the EER as a function of adaptation iteration, when adapting on non-password phrases for a single GMM or PCBV solution and testing on password phrases. It can be seen that the GMM solution is very sensitive to adaptation on non-password phrases,

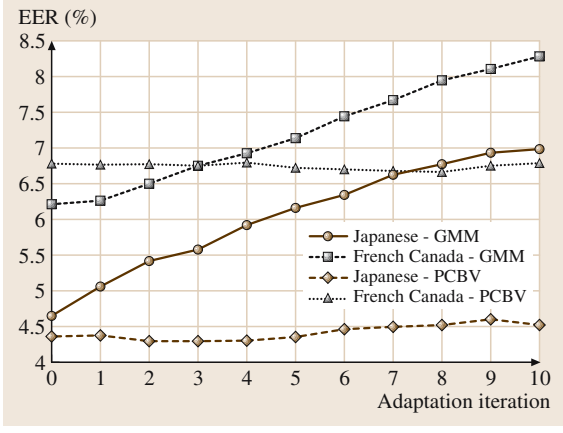


Fig. 37.9 The effect of speaker model adaptation with non-password phrases on the accuracy of password phrases (EER). Adaptation iteration 0 is the enrollment session. The experiments were carried out on over 24 k attempts from genuine speaker and imposters. Based on the lower number of trials (genuine in our case), the 90% confidence interval on the measures is 0.3%

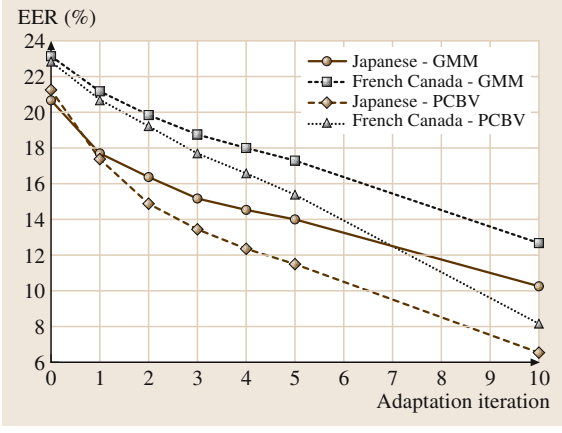


Fig. 37.10 The effect of speaker model adaptation with non-password phrases on the accuracy of non-password phrases (EER). Adaptation iteration 0 is the enrollment session. The experiments were carried out on over 14 k attempts from genuine speaker and imposters. Based on the lower number of trials (genuine in our case), the 90% confidence interval on the measures is 0.5%

Table 37.6 The measured FA rate using an automatic impostor trial generation algorithm for different conditions and data sets. Note that the target FA rate was 1.0%

Experimental setup	English_US_1	English_US_2	English_UK_1	Average
[37.45] algorithm without offset	2.10%	1.15%	2.39%	1.88%
[37.45] algorithm with 0.15 offset	1.13%	0.81%	1.35%	1.10%
New binning without offset	0.86%	0.75%	1.25%	0.95%

whereas the PCBV is not. This is due to the fact that PCBV uses alignments from a speech recognition engine to segregate frames into different modeling units while the GMM does not: this leads to smearing of the speaker model in the case of the GMM solution. Figure 37.10 shows the improvements in the accuracy on non-password phrases in the same context. Note that iterations 1–5 do not have overlapping phrases with the testing lexicon: iteration 10 has some overlap, which is not unrealistic from a speech application point of view. As expected, the accuracy of the non-password phrase is improved by the adaptation process for both GMM and PCBV, with a much greater improvement for PCBV. After 10 adaptation iterations, the accuracy is 6–8% EER (and has not yet reached a plateau), which makes this solution a viable solution. It can also be noted that PCBV with adaptation on non-password phrases improves the accuracy faster than its single-GMM counterpart, taking half the adaptation iterations to achieve a similar EER (Fig. 37.10). In summary, speaker model adaptation and PCBV form the basis for delivering stable accuracy on password phrases while dramatically improving the accuracy for non-password phrases. This set of results is another illustration of the power of speaker model adaptation and represents one possible implementation for protection against recordings. Any improvement in this area is important for the text-dependent speaker recognition field as well as commercial applications.

37.3.7 Automatic Impostor Trials Generation

As mentioned above, application developers usually want to know how secure their speech application is. Usually, the design of the security layer is based on the choice of the password phrase, the choice of the enrollment and verification dialogs, and the security level (essentially the FA rate). From these decisions follow the FR and RR rates. Using off-the-shelf threshold settings will usually only give a range of target FA rates, but will rarely give any hint on the FR and RR for the current designed dialog [37.36]. Often application developers want a realistic picture of the accuracy of their system (FA, FR, and RR) based on their data. Since the FA rate is important, this has to be measured with a high degree of confidence. To do this, one requires a tremendous amount of data. As an example, to measure an FA of $1\% \pm 0.3\%$ nine times out of ten, 3000 imposter trials are required [37.1]. For a higher degree of precision such as $\pm 0.1\%$, more than 30 000 imposter trials are needed. Collecting data for imposter trials results in a lot of issues; it is costly, requires data management and tagging, cannot really be done on production systems if adaptation of speaker models is enabled, etc. However, collecting genuine speaker attempts can be done simply by archiving utterances and the associated claimed identity; depending on the traffic, a lot of data can be gathered quickly. Note that some manual tagging may be required to flag true imposter attempts – usually low-scoring genuine speaker attempts. The data gathered is also valuable because it can come from the production system.

For password phrases that are common to all users of a system, generating imposter attempts is easy once the data has been collected and tagged: it can be done using a round-robin. However, if the password phrase is unique for each genuine speaker, a round-robin cannot be used. In this case, the lexical content of the imposter attempts will be mismatched to the target speaker models, the resulting attempt will be grossly unchallenging and will lead to underestimation of the actual FA rate. In [37.45], an algorithm to estimate the FA rate accurately using only genuine speaker attempts was presented. The essence of the idea is to use a round-robin for imposter trial generation, but to quantify the amount of lexical mismatch between the attempt and target speaker model. Each imposter attempt will have a lexical mismatch value associated with it. This can be thought of as a lexical distance (mismatch) between two strings. Intuitively, we want the following order for the lexical mismatch value with respect to the target string $1234 : 1234 < 1256 < 1526 < 5678$. Note that [37.45] and the following are based on digit strings, but can easily be applied to general text by using phonemes as the atom instead of digits. A variant of the Levenstein distance was used to bin imposter attempts. For each bin,

the threshold to achieve the target FA rate was calculated. A regression between the Levenstein distance and threshold for the target FA is used to extrapolate the operational threshold for the target FA rate. For the development of this algorithm, three test sets from data collections and trials were used. These had a set of *real* impostor attempts that we used to assess the accuracy of the algorithm. The first line of Table 37.6 shows the *real* FA rate measured at the operational threshold as calculated by the algorithm above. In [37.45], to achieve good accuracy, an offset of 0.15 needed to be introduced (the second line in the table). The algorithm had one free parameter. It was later noticed that, within a bin with a given Levenstein distance, some attempts were more competitive than others. For example, the target/attempt pairs 97 526/97 156 and 97 526/97 756 had the same Levenstein distance. However, the second pair is more competitive because all of the digits in the attempt are present in the target and hence have been seen during the enrollment. A revised binning was performed and is presented as the last line in Table 37.6. The average measured FA rate is much closer to the target FA rate and this revised algorithm does not require any free parameters.

Once the threshold for the desired FA rate has been calculated, it is simple to extract the FR and RR rates from the same data. Reducing the cost of deployment is critical for making speaker recognition a mainstream biometric technique. Any advances in this direction is thus important.

37.4 Concluding Remarks

This chapter on text-dependent speaker recognition has been designed to illustrate the current technical challenges of the field. The main challenges are robustness to channel and lexical mismatches. Several results were presented to illustrate these two key challenges under a number of conditions. Adaptation of the speaker models yields advantages to address these challenges but this needs to be properly engineered to be deployable on a large scale while maintaining a stable operating point. Several new research avenues were reviewed.

When relevant, parallels between the text-dependent and text-independent speaker recognition fields were drawn. The distinctions between the two fields becomes thin when considering the work by *Sturim* et al. [37.2] and text-dependent speaker recognition with heavy lexical mismatch, as described in Sect. 37.3.6. This research area should provide a very fertile ground for future advances in the speaker recognition field.

Finally, special care was taken to illustrate, using relevant (live or trial) data, the specific challenges facing text-dependent speaker recognition in actual deployment situations.

References

37.1 A. Martin, M. Przybocki, G. Doddington, D.A. Reynolds: The NIST speaker recognition evaluation – Overview, methodology, systems, results, perspectives, Speech Commun. **31**, 225–254 (2000)

37.2 D.E. Sturim, D.A. Reynolds, R.B. Dunnk, T.F. Quatieri: Speaker verification using text-constrained gaussian mixture models, Proc. IEEE ICASSP **2002**(1), 677–680 (2002)

37.3 K. Boakye, B. Peskin: Text-constrained speaker recognition on a text-independent task, Proc. Odyssey Speaker Recognition Workshop, Vol. 2004 (2004)

37.4 D. Boies, M. Hébert, L.P. Heck: Study of the effect of lexical mismatch in text-dependent speaker verification, Proc. Odyssey Speaker Recognition Workshop, Vol. 2004 (2004)

37.5 M. Wagner, C. Summerfield, T. Dunstone, R. Summerfield, J. Moss: An evaluation of commercial off-the-shelf speaker verification systems, Proc. Odyssey Speaker Recognition Workshop, Vol. 2006 (2006)

37.6 A. Higgins, L. Bahler, J. Porter: Speaker verification using randomized phrase prompting, Digit. Signal Process. **1**, 89–106 (1991)

37.7 M.J. Carey, E.S. Parris, J.S. Briddle: A speaker verification system using alpha-nets, Proc. IEEE ICASSP, Vol. 1981 (1981) pp. 397–400

37.8 L.P. Heck, M. Weintraub: Handset dependent background models for robust text-independent

speaker recognition, Proc. IEEE ICASSP **1997**(2), 1037–1040 (1997)
37.9 A.E. Rosenberg, S. Parthasarathy: The use of cohort normalized scores for speaker recognition, Proc. IEEE ICASSP **1996**(1), 81–84 (1996)
37.10 C. Barras, J.-L. Gauvain: Feature and score normalization for speaker verification of cellular data, Proc. IEEE ICASSP **2003**(2), 49–52 (2003)
37.11 Y. Liu, M. Russell, M. Carey: The role of dynamic features in text-dependent and -independent speaker verification, Proc. IEEE ICASSP **2006**(1), 669–672 (2006)
37.12 D. Reynolds: Channel robust speaker verification via feature mapping, Proc. IEEE ICASSP **2003**(2), 53–56 (2003)
37.13 R. Teunen, B. Shahshahani, L.P. Heck: A model-based transformational approach to robust speaker recognition, Proc. ICSLP **2000**(2), 495–498 (2000)
37.14 R.O. Duda, P.E. Hart, D.G. Stork: *Pattern Classification*, 2nd edn. (Wiley, New York 2001)
37.15 T. Kato, T. Shimizu: Improved speaker verification over the cellular phone network using phoneme-balanced and digit-sequence-preserving connected digit patterns, Proc. IEEE ICASSP **2003**(2), 57–60 (2003)
37.16 T. Matsui, S. Furui: Concatenated phoneme models for text-variable speaker recognition, Proc. IEEE ICASSP **1993**(2), 391–394 (1993)
37.17 S. Parthasarathy, A.E. Rosenberg: General phrase speaker verification using sub-word background models and likelihood ratio scoring, Proc. ICSLP **1996**(4), 2403–2406 (1996)
37.18 C.W. Che, Q. Lin, D.S. Yuk: An HMM approach to text-prompted speaker verification, Proc. IEEE ICASSP **1996**(2), 673–676 (1996)
37.19 M. Hébert, L.P. Heck: Phonetic class-based speaker verification, Proc. Eurospeech, Vol. 2003 (2003) pp. 1665–1668
37.20 E.G. Hansen, R.E. Slygh, T.R. Anderson: Speaker recognition using phoneme-specific GMMs, Proc. Odyssey Speaker Recognition Workshop, Vol. 2004 (2004)
37.21 M. Schmidt, H. Gish: Speaker identification via support vector classifiers, Proc. IEEE ICASSP **1996**(1), 105–108 (1996)
37.22 W.M. Campbell, D.E. Sturim, D.A. Reynolds, A. Solomonoff: SVM based speaker verification using a GMM supervector kernel and NAP variability compensation, Proc. IEEE ICASSP **2006**(1), 97–100 (2006)
37.23 N. Krause, R. Gazit: SVM-based speaker classification in the GMM model space, Proc. Odyssey Speaker Recognition Workshop, Vol. 2006 (2006)
37.24 S. Fine, J. Navratil, R.A. Gopinath: A hybrid GMM/SVM approach to speaker identification, Proc. IEEE ICASSP **2001**(1), 417–420 (2001)
37.25 W.M. Campbell: A SVM/HMM system for speaker recognition, Proc. IEEE ICASSP **2003**(2), 209–212 (2003)
37.26 S. Furui: Cepstral analysis techniques for automatic speaker verification, IEEE Trans. Acoust. Speech **29**, 254–272 (1981)
37.27 V. Ramasubramanian, A. Das, V.P. Kumar: Text-dependent speaker recognition using one-pass dynamic programming algorithm, Proc. IEEE ICASSP **2006**(2), 901–904 (2006)
37.28 A. Sankar, R.J. Mammone: Growing and pruning neural tree networks, IEEE Trans. Comput. **42**, 272–299 (1993)
37.29 K.R. Farrell: Speaker verification with data fusion and model adaptation, Proc. ICSLP **2002**(2), 585–588 (2002)
37.30 D.A. Reynolds, T.F. Quatieri, R. B.Dunn: Speaker verification using adapted gaussian mixture models, Digit. Signal Process. **10**, 19–41 (2000)
37.31 J.-L. Gauvain, C.-H. Lee: Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains, IEEE T. Speech Audi. Process. **2**, 291–298 (1994)
37.32 D.A. Reynolds: Comparison of background normalization methods for text-independent speaker verification, Proc. EuroSpeech **1997**(2), 963–966 (1997)
37.33 N. Mirghafori, L.P. Heck: An adaptive speaker verification system with speaker dependent a priori decision thresholds, Proc. ICSLP **2002**(2), 589–592 (2002)
37.34 R. Auckenthaler, M.J. Carey, H. Lloyd-Thomas: Score normalization for text-independent speaker verification systems, Digit. Signal Process. **10**, 42–54 (2000)
37.35 M. Hébert, D. Boies: T-Norm for text-dependent commercial speaker verification applications: effect of lexical mismatch, Proc. IEEE ICASSP **2005**(1), 729–732 (2005)
37.36 N. Mirghafori, M. Hébert: Parametrization of the score threshold for a text-dependent adaptive speaker verification system, Proc. IEEE ICASSP **2004**(1), 361–364 (2004)
37.37 T. Matsui, T. Nishitani, S. Furui: Robust methods for updating model and a priori threshold in speaker verificaiton, Proc. IEEE ICASSP, Vol. 1996 (1996) pp. 97–100
37.38 C. Barras, S. Meignier, J.-L. Gauvain: Unsupervised online adaptation for speaker verification over the telephone, Proc. Odyssey Speaker Recognition Workshop, Vol. 2004 (2004)
37.39 C. Fredouille, J. Mariéthoz, C. Jaboulet, J. Hennebert, J.-F. Bonastre, C. Mokbel, F. Bimbot:

Behavior of a bayesian adaptation method for incremental enrollment in speaker verification, Proc. IEEE ICASSP, Vol. 2000 (2000)
37.40 L.P. Heck, N. Mirghafori: Online unsupervised adaptation in speaker verification, Proc. ICSLP, Vol. 2000 (2000)
37.41 L.P. Heck: On the deployment of speaker recognition for commercial applications, Proc. Odyssey Speaker Recognition Workshop, Vol. 2004 (2004), keynote speech
37.42 K. Wadhwa: Voice verification: technology overview anf accuracy testing results, Proc. Biometrics Conference, Vol. 2004 (2004)
37.43 M.J. Carey, R. Auckenthaler: User validation for mobile telephones, Proc. IEEE ICASSP, Vol. 2000 (2000)
37.44 L.P. Heck, D. Genoud: Integrating speaker and speech recognizers: automatic identity claim capture for speaker verification, Proc. Odyssey Speaker Recognition Workshop, Vol. 2001 (2001)
37.45 M. Hébert, N. Mirghafori: Desperately seeking impostors: data-mining for competitive impostor testing in a text-dependent speaker verification system, Proc. IEEE ICASSP **2004**(2), 365–368 (2004)
37.46 T.F. Quatieri, E. Singer, R.B. Dunn, D.A. Reynolds, J.P. Campbell: Speaker and language recognition using speech codec partameters, Proc. EuroSpeech, Vol. 1999 (1999) pp. 787–790
37.47 L.P. Heck, Y. Konig, M.K. Sönmez, M. Weintraub: Robustness to telephone handset distortion in speaker recognition by discriminative feature design, Speech Commun. **31**, 181–192 (2000)
37.48 M. Siafarikas, T. Ganchev, N. Fakotakis, G. Kokkinakis: Overlapping wavelet packet features for speaker verification, Proc. EuroSpeech, Vol. 2005 (2005)
37.49 D. Reynolds: Speaker identification and verification using Gaussian mixture speaker models, Speech Commun. **17**, 91–108 (1995)
37.50 O. Siohan, C.-H. Lee, A.C. Surendran, Q. Li: Background model design for flexible and portable speaker verification systems, Proc. IEEE ICASSP **1999**(2), 825–829 (1999)
37.51 L.P. Heck, N. Mirghafori: Unsupervised on-line adaptation in speaker verification: confidence-based updates and improved parameter estimation, Proc. Adaptation in Speech Recognition, Vol. 2001 (2001)
37.52 D. Hernando, J.R. Saeta, J. Hernando: Threshold estimation with continuously trained models in speaker verification, Proc. Odyssey Speaker Recognition Workshop, Vol. 2006 (2006)
37.53 A. Sankar, A. Kannan: Automatic confidence score mapping for adapted speech recognition systems, Proc. IEEE ICASSP **2002**(1), 213–216 (2002)
37.54 D. Genoud, G. Chollet: Deliberate imposture: a challenge for automatic speaker verification systems, Proc. EuroSpeech, Vol. 1999 (1999) pp. 1971–1974
37.55 B.L. Pellom, J.H.L. Hansen: An experimental study of speaker verification sensitivity to computer voice-altered imposters, Proc. IEEE ICASSP **1999**(2), 837–840 (1999)
37.56 D. Matrouf, J.-F. Bonastre, C. Fredouille: Effect of speech transformation on impostor acceptance, Proc. IEEE ICASSP **2006**(2), 933–936 (2006)

38. Text-Independent Speaker Recognition

D. A. Reynolds, W. M. Campbell

In this chapter, we focus on the area of text-independent speaker verification, with an emphasis on unconstrained telephone conversational speech. We begin by providing a general likelihood ratio detection task framework to describe the various components in modern text-independent speaker verification systems. We next describe the general hierarchy of speaker information conveyed in the speech signal and the issues involved in reliably exploiting these levels of information for practical speaker verification systems. We then describe specific implementations of state-of-the-art text-independent speaker verification systems utilizing low-level spectral information and high-level token sequence information with generative and discriminative modeling techniques. Finally, we provide a performance assessment of these systems using the National Institute of Standards and Technology (NIST) speaker recognition evaluation telephone corpora.

38.1 Introduction

With the merging of telephony and computer networks, the growing use of speech as a modality in man–machine communication, and the need to manage ever increasing amounts of recorded speech in audio archives and multimedia applications, the utility of recognizing a person from his or her voice is increasing. While the area of speech recognition is concerned with extracting the linguistic message underlying a spoken utterance, speaker recognition is concerned with extracting the identity of the person speaking the utterance. Applications of speaker recognition are wide ranging, including: facility or computer access control [38.1,2], telephone voice authentication for long-distance calling or banking access [38.3], intelligent answering machines with personalized caller greetings [38.4], and automatic speaker labeling of recorded meetings for speaker-dependent audio indexing (speech skimming) [38.5, 6].

Depending upon the application, the general area of speaker recognition is divided into two specific tasks: identification and verification. In speaker identification, the goal is to determine which one of a group of known voices best matches the input voice sample. This is also referred to as *closed-set* speaker identification. Applications of pure closed-set identification are limited to cases where only enrolled speakers will be encountered, but it is a useful means of examining the separability of speakers' voices or finding similar sounding speakers, which has applications in speaker-adaptive speech recognition. In verification, the goal is to determine from a voice sample if a person is who he or she claims to be. This is sometimes referred to as the *open-set* problem, because this task requires distinguishing a claimed speaker's voice known to the system from a potentially large group of voices unknown to the system (i. e., impostor speakers). Verification is the basis

for most speaker recognition applications and the most commercially viable task. The merger of the closed-set identification and open-set verification tasks, called open-set identification, performs like closed-set identification for known speakers but must also be able to classify speakers unknown to the system into a *none of the above* category.

These tasks are further distinguished by the constraints placed on the speech used to train and test the system and the environment in which the speech is collected [38.7]. In a text-dependent system, the speech used to train and test the system is constrained to be the same word or phrase. In a text-independent system, the training and testing speech are completely unconstrained. Between text dependence and text independence, a vocabulary-dependent system constrains the speech to come from a limited vocabulary, such as the digits, from which test words or phrases (e.g., digit strings) are selected. Furthermore, depending upon the amount of control allowed by the application, the speech may be collected from a noise-free environment using a wide-band microphone or from a noisy, narrow-band telephone channel.

In this chapter, we focus on the area of text-independent speaker verification, with an emphasis on unconstrained telephone conversational speech. While many of the underlying algorithms employed in text-independent and text-dependent speaker verification are similar, text-independent applications have the additional challenge of operating unobtrusively to the user with little to no control over the user's behavior (i. e., the user is speaking for some other purpose, not to be verified, so will not cooperate to speak more clearly, use a limited vocabulary or repeat phrases). Further, the ability to apply text-independent verification to unconstrained speech encourages the use of audio recorded from a wide variety of sources (e.g., speaker indexing of broadcast audio or forensic matching of law-enforcement microphone recordings), emphasizing the need for compensation techniques to handle variable acoustic environments and recording channels.

This chapter is organized as follows. We begin by providing a general likelihood ratio detection task framework to describe the various components in modern text-independent speaker verification systems. We next describe the general hierarchy of speaker information conveyed in the speech signal and the issues involved in reliably exploiting these levels of information for practical speaker verification systems. We then describe specific implementations of state-of-the-art text-independent speaker verification systems utilizing low-level spectral information and high-level token sequence information with generative and discriminative modeling techniques. Finally we provide a performance assessment of these systems using the NIST speaker recognition evaluation telephone corpora.

38.2 Likelihood Ratio Detector

Given a segment of speech, Y, and a hypothesized speaker, S, the task of speaker detection, also referred to as verification, is to determine if Y was spoken by S. An implicit assumption often used is that Y contains speech from only one speaker. Thus, the task is better termed single-speaker detection. If there is no prior information that Y contains speech from a single speaker, the task becomes multispeaker detection. In this paper we will focus on the core single-speaker detection task. Discussion of systems that handle the multispeaker detection task can be found in [38.8].

The single-speaker detection task can be restated as a basic hypothesis test between

H_0 : Y is from the hypothesized speaker S

H_1 : Y is *not* from the hypothesized speaker S .

From statistical detection theory, the optimum test to decide between these two hypotheses is a likelihood ratio test given by

$$\frac{p(Y \mid H_0)}{p(Y \mid H_1)} \begin{cases} \geq \theta & \text{accept } H_0 \\ < \theta & \text{reject } H_0 \end{cases}, \tag{38.1}$$

where $p(Y \mid H_i), i = 0, 1$, is the probability density function for the hypothesis H_i evaluated for the observed speech segment Y, also referred to as the *likelihood* of the hypothesis H_i given the speech segment ($p(A \mid B)$ is referred to as a likelihood when B is considered the independent variable in the function). Strictly speaking, the likelihood ratio test is only optimal when the likelihood functions are known exactly, which is rarely the case. The decision threshold for accepting or rejecting H_0 is θ. Thus, the basic goal of a speaker detection system is to determine techniques to compute the likelihood ratio between the two likelihoods, $p(Y \mid H_0)$ and $p(Y \mid H_1)$. Depending upon the techniques used, these likelihoods

are explicitly or implicitly modeled and applied in the speaker detection systems.

Figure 38.1 shows the basic components found in speaker detection systems. The role of the front-end processing is to extract features from the speech signal that convey speaker-dependent information. In addition, techniques to minimize non-speaker-dependent effects from these features, such as linear filtering or additive noise, are also employed in the front-end processing. The output of this stage is a sequence of feature vectors representing the test segment, $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$, where $\boldsymbol{x}_t$ is a feature vector indexed at discrete time $t \in [1, 2, \ldots, T]$. Features can be continuous or discrete with multiple dimensions, may be extracted at asynchronous time instances and can cover variable temporal durations. For example, the sequence of words spoken as estimated by a speech recognizer (series of one-dimensional discrete, asynchronous, variable temporal duration values), or the overall average pitch and energy in an utterance (single two-dimensional continuous-value vector) could be used as features. Thus, features can be extracted that capture short- and long-span speaker characteristics. Common features used in text-independent speaker detection systems are described in Sect. 38.3.

These feature vectors are then used to compute a score to evaluate the likelihood ratio test between H_0 and H_1. The exact way in which the hypothesis likelihood functions are represented and scored against the feature vectors depends on the classifier employed. The parameters of the classifier are obtained from the speaker enrollment or training phase, which is discussed in Sect. 38.4.

In generative modeling approaches, the likelihoods are explicitly represented by probability density models. Mathematically, H_0 is represented by a model denoted λ_{hyp}, which characterizes the hypothesized speaker S in the feature space of $\boldsymbol{x}$. For example, one could assume that the feature vectors for H_0 were generated from a Gaussian distribution so that λ_{hyp} would denote the mean vector and covariance matrix parameters of a Gaussian distribution. The alternative hypothesis, H_1, is similarly represented by the model $\lambda_{\overline{\text{hyp}}}$. The likelihood ratio statistic (or score) is then $p(X \mid \lambda_{\text{hyp}})/p(X \mid \lambda_{\overline{\text{hyp}}})$. Often, the logarithm of this statistic is used giving the log-likelihood ratio (LLR) score for the utterance,

$$\Lambda(X) = \log p(X \mid \lambda_{\text{hyp}}) - \log p(X \mid \lambda_{\overline{\text{hyp}}}) \,. \qquad (38.2)$$

In discriminative modeling approaches such as support vector machines (SVMs), the two likelihoods are not explicitly modeled separately, but a detection score similar to the log likelihood ratio score is computed (see Sect. 38.4.2). Although not precise mathematically, it is useful to use this likelihood ratio framework for both generative and discriminative approaches since it emphasizes the important roles of the speaker and alternative hypotheses in the modeling and detection process.

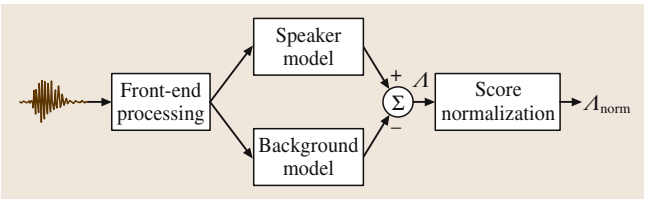


Fig. 38.1 Likelihood-ratio-based speaker detection system

After a likelihood ratio score is produced for the test speech and hypothesized speaker, it is often further processed by some form of score normalization and/or calibration. Despite mitigation steps taken during feature extraction (see Sect. 38.3.1) and model parameter estimation (see Sect. 38.4), the LLR score produced will be influenced by several nonspeaker factors, such as the spoken text in the test speech, differences in the microphones used and acoustic environment of the enrollment and test speech, and inaccuracies and assumptions in the models applied. Score normalization aims to compensate for these variabilities by removing score biases and scale factors estimated during enrollment. Several approaches have been proposed for score normalization, for example, Z-norm [38.9], H-norm [38.10], and T-norm [38.11], which compensate the LLR score as

$$\Lambda^{\text{norm}}(X) = \frac{\Lambda(X) - \mu(X, S)}{\sigma(X, S)} \,. \qquad (38.3)$$

The main difference in these score normalization techniques is how the biases, $\mu(X, S)$, and scale factors, $\sigma(X, S)$, are estimated. For example, in the H-norm, handset-microphone-dependent values are estimated, while in the T-norm scores from other speaker models are used to estimate these bias and scale factors.

A further step applied after or as part of score normalization is that of score calibration or confidence estimation. The aim in this process is to convert the LLR score values into *human-interpretable* values, such as posterior probabilities, with a defined range of, for example, (0–1). Confidence estimation techniques range from simple formula-based approaches that merely convert the LLR score to a posterior probability using a given prior probability, to more-sophisticated ap-

proaches that integrate in external signal measurement information, such as signal-to-noise measures, with the LLR score using artificial neural networks [38.12].

With this framework, we next describe specific examples of these components for modern text-independent speaker detection systems.

38.3 Features

One of the fundamental components in any speaker detection system is the features extracted from the speech signal that convey information about the speaker's identity. Unfortunately there is no single attribute of speech that conveys identity, rather it is conveyed through several different types or levels of information in the speech signal. These levels can roughly be categorized into a hierarchy running from low-level information, such as the sound of a person's voice, related to physical traits of the vocal apparatus, to high-level information, such as particular word usage (idiolect), related to learned habits, dialect, and style.

Features related to low-level information based on short-term spectral analysis are the dominant type used in text-independent speaker detection systems. This is primarily due to the computational ease of extracting these features and their proven effectiveness in terms of detection performance.

In recent years, there has been increased interest in exploiting high-level features for text-independent speaker detection systems. This has been spurred on by the continual advancement of phone and speech recognition systems used to extract features for high-level characterization. Work done at the 2002 SuperSID workshop [38.13] and subsequent related efforts demonstrated how systems using high- and low-level features could be successfully fused to improve overall accuracy. While high-level features are a promising new area of research in text-independent speaker recognition, they often have substantial computational costs (e.g., running a large-vocabulary speech recognizer), which need to be balanced against relatively modest accuracy gains.

38.3.1 Spectral Features

Several processing steps occur in the front-end analysis to extract spectral-based features. First, the speech is segmented into frames by a 20 ms window progressing at a 10 ms frame rate. A speech activity detector is then used to discard silence/noise frames. Typically, the speech detector discards 20–25% of the signal from conversational telephone speech.

Next, mel-scale cepstral feature vectors are extracted from the speech frames. The mel-scale cepstrum is the discrete cosine transform of the log-spectral energies of the speech segment Y. The spectral energies are calculated over logarithmically spaced filters with increasing bandwidths (*mel-filters*). A detailed description of the feature extraction steps can be found in [38.14]. For bandlimited telephone speech, cepstral analysis is usually performed only over the mel-filters in the telephone pass band (300–3400 Hz). All cepstral coefficients except the zeroth value (the DC level of the log-spectral energies) are retained in the processing. Lastly, delta-cepstra values are computed using a first-order derivative of an orthogonal polynomial temporal fit over ± 2 feature vectors (two to the left and two to the right over time) from the current vector [38.15].

Finally, the feature vectors are channel-normalized to remove linear channel convolutional effects. With cepstral features, convolutional effects appear as additive biases. Both cepstral mean subtraction (CMS) and RASTA filtering [38.16] have been used successfully and, in general, both methods have comparable performance for single-speaker detection tasks. When training and recognition speech are collected from different microphones or channels (e.g., different telephone handsets and/or lines), this is a crucial step for achieving good recognition accuracy. However, this linear compensation does not completely eliminate the performance loss under mismatched microphone conditions, so more-sophisticated compensation techniques such as feature mapping [38.17], where transformations are trained to map features coming from particular channels (e.g., microphones) into channel independent features, are also applied.

There are many variants of these cepstral features, such as using linear prediction coding (LPC) spectral analysis instead of fast Fourier transform (FFT)-based spectral analysis, but the basic steps are similar and performance does not vary significantly.

38.3.2 High-Level Features

For high-level feature extraction, input speech is converted into a series of tokens, $T = \{t_i\}$. The tokens are time-ordered discrete symbols and represent linguistically significant interpretations of the input signal.

Examples of token types are words, phones, and pitch gestures.

High-level feature extraction involves many tradeoffs. First, designing token extraction systems is, in general, a difficult process. If the high-level features are specific to the language of interest, then significant effort may be required to implement the system. For example, speech-to-text (STT) systems such as the BBN Byblos system [38.18] require training using large corpora in the language of interest and significant engineering effort to produce high-accuracy output. A second consideration in token extraction is how it will be used with other speaker recognition recognition systems. Some token systems, such as STT, have been shown to provide significant fusion gains when combined with standard acoustic systems [38.19], but have low speaker recognition accuracy alone. Other token systems, such as phone token [38.20], have excellent standalone accuracy, but do not provide significant complimentary information to standard spectral methods. A third consideration for token system selection is length of enrollment and verification. A token system that produces tokens at a low rate, e.g., words, requires multi-conversation enrollment to produce good results. Other higher rate tokens, e.g., phones, require less enrollment for equivalent accuracy.

Modeling of the token stream is usually accomplished by computing probabilities of n-grams of token contexts [38.21] and then applying a classifier. We discuss two methods based on SVMs and standard likelihood ratio techniques in Sect. 38.4.3.

38.4 Classifiers

In this section, we describe popular and successful generative and discriminative classifiers used for both high- and low-level features. First we present spectral-feature-based classifiers using adapted Gaussian mixture models and sequence-based support vector machines. This is followed by high-level (token sequence) feature based n-gram and SVM classifiers.

38.4.1 Adapted Gaussian Mixture Models

Gaussian Mixture Models

An important step in the implementation of the likelihood ratio detector of (38.2) is selection of the actual likelihood function, $p(X|\lambda)$. The choice of this function is largely dependent on the features being used as well as specifics of the application. For text-independent speaker recognition using continuous features, where there is no prior knowledge of what the speaker will say, the most successful likelihood function has been Gaussian mixture models (GMMs). In text-dependent applications, where there is strong prior knowledge of the spoken text, additional temporal knowledge can be incorporated by using hidden Markov models (HMMs) as the basis for the likelihood function. To date, however, use of more-complicated likelihood functions, such as those based on HMMs, have shown no advantage over GMMs for text-independent speaker detection tasks. As more-accurate knowledge of the spoken text is gained via speech recognition, lessening the distinction between text-dependent and text-independent tasks, this may change.

For a D-dimensional feature vector, $\boldsymbol{x}$, the mixture density used for the likelihood function is defined as

$$p(\boldsymbol{x}|\lambda) = \sum_{i=1}^{M} w_i \; p_i(\boldsymbol{x}) \; . \tag{38.4}$$

The density is a weighted linear combination of M unimodal Gaussian densities, $p_i(\boldsymbol{x})$, each parameterized by a mean $D \times 1$ vector, $\boldsymbol{\mu}_i$, and a $D \times D$ covariance matrix, Σ_i;

$$p_i(\boldsymbol{x}) = \frac{1}{(2\pi)^{D/2} |\Sigma_i|^{1/2}} \times \exp\left[-\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu}_i)^{\mathrm{T}} \; \Sigma_i^{-1} \; (\boldsymbol{x} - \boldsymbol{\mu}_i) \right] . \tag{38.5}$$

The mixture weights, w_i, furthermore satisfy the constraint $\sum_{i=1}^{M} w_i = 1$. Collectively, the parameters of the density model are denoted by $\lambda = \{w_i, \boldsymbol{\mu}_i, \Sigma_i\}$, where $i = 1, \ldots, M$.

While the general model form supports full covariance matrices, i. e., a covariance matrix with all its elements, usually only diagonal covariance matrices are used. This is done for three reasons. First, the density modeling of an M^{th}-order full covariance GMM can equally well be achieved using a larger-order diagonal covariance GMM. GMMs with $M > 1$ using diagonal covariance matrices can model distributions of feature vectors with correlated elements. Only in the degenerate case of $M = 1$ is the use of a diagonal covariance matrix incorrect for feature vectors with correlated elements. Second, diagonal-matrix

GMMs are more computationally efficient than full-covariance GMMs for training since repeated inversions of a $D \times D$ matrix are not required. Third, empirically we have observed that diagonal-matrix GMMs outperform full-matrix GMMs.

Given a collection of training vectors, maximum-likelihood model parameters are estimated using the iterative expectation-maximization (EM) algorithm [38.22]. The EM algorithm iteratively refines the GMM parameters to increase the likelihood of the estimated model for the observed feature vectors monotonically, i. e., for iterations k and $k+1$, $p(X|\lambda^{(k+1)}) > p(X|\lambda^{(k)})$. Generally, five iterations are sufficient for parameter convergence. The EM equations for training a GMM can be found in [38.23].

The feature vectors of X are assumed independent, so the log-likelihood of a model λ for a sequence of feature vectors, $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$, is computed as

$$\log p(X \mid \lambda) = \sum_{t=1}^{\mathrm{T}} \log p(\boldsymbol{x}_t \mid \lambda) \,, \tag{38.6}$$

where $p(\boldsymbol{x}_t \mid \lambda)$ is computed as in (38.4). Often, the average log-likelihood value is used, by dividing $\log p(X \mid \lambda)$ by T. This is done to normalize out duration effects from the log-likelihood value. Since the incorrect assumption of independence is underestimating the actual likelihood value with dependencies, this scaling factor can also be considered a rough compensation factor to the likelihood value in (38.6).

The GMM can be viewed as a hybrid between a parametric and nonparametric density model. Like a parametric model it has structure and parameters that control the behavior of the density in known ways, but without constraints that the data must be of a specific distribution type, such as Gaussian or Laplacian. Like a nonparametric model, the GMM has many degrees of freedom to allow arbitrary density modeling, without undue computation and storage demands. It can also be thought of as a single-state HMM with a Gaussian mixture observation density, or an ergodic Gaussian observation HMM with fixed, equal transition probabilities. Here, the Gaussian components can be considered to be modeling the underlying broad phonetic sounds that characterize a person's voice. A more-detailed discussion of how GMMs apply to speaker modeling can be found in [38.24].

The advantages of using a GMM as the likelihood function are that it is computationally inexpensive, is based on a well-understood statistical model, and, for text-independent tasks, is insensitive to the temporal aspects of the speech, modeling only the underlying distribution of acoustic observations from a speaker. The latter is also a disadvantage in that higher-levels of information about the speaker conveyed in the temporal speech signal are not used.

Universal Background Model

We next describe how the GMM described above is used to build the likelihood ratio detector in (38.2). While the model for H_0, λ_{hyp}, is well defined and is estimated using training speech from S, the model for the alternative hypothesis, $\lambda_{\overline{\mathrm{hyp}}}$, is less well defined since it potentially must represent the entire space of possible alternatives to the hypothesized speaker. Two main approaches have been taken for this alternative hypothesis modeling. The first approach is to use a set of other speaker models to cover the space of the alternative hypothesis. In various contexts, this set of other speakers has been called likelihood ratio sets [38.2], cohorts [38.25], and background speakers [38.23]. Given a set of N background speaker models $\{\lambda_1, \ldots, \lambda_N\}$, the alternative hypothesis model is represented by

$$p(X \mid \lambda_{\overline{\mathrm{hyp}}}) = \mathcal{F}[p(X \mid \lambda_1), \ldots, p(X \mid \lambda_N)], \tag{38.7}$$

where $\mathcal{F}()$ is some function, such as the average or maximum, of the likelihood values from the background speaker set. The selection, size, and combination of the background speakers has been the subject of much research. In general, it has been found that to obtain the best performance with this approach requires the use of speaker-specific background speaker sets. This can be a drawback in an applications using a large number of hypothesized speakers, each requiring their own background speaker set.

The second major approach to alternative hypothesis modeling is to pool speech from several speakers and train a single model. Various terms for this single model are a general model [38.26], a world model, and a universal background model [38.27]. Given a collection of speech samples from a large number of speakers representative of the population of speakers expected during recognition, a single model λ_{bkg} is trained to represent the alternative hypothesis. Research on this approach has focused on selection and composition of the speakers and speech used to train the single model. The main advantage of this approach is that a single speaker-independent model can be trained once for a particular task and then used for all hypothesized speakers in that task. It is also possible to use multiple background models tailored to specific sets of speakers. Overall,

the use of a single background model [which we will refer to as the universal background model (UBM)], is the dominant approach used in text-independent speaker detection systems.

The UBM is a large GMM trained to represent the speaker-independent distribution of features. Specifically, we want to select speech that is reflective of the expected alternative speech to be encountered during recognition. This applies to both the type and quality of speech, as well as the composition of speakers. For example, in the NIST speaker recognition evaluation (SRE) single-speaker detection tests, it is known a priori that the speech comes from local and long-distance telephone calls, and that male hypothesized speakers will only be tested against male speech. In this case, we would train the UBM used for male tests using only male telephone speech. In the case where there is no prior knowledge of the gender composition of the alternative speakers, we would train using gender-independent speech.

Other than these general guidelines and experimentation, there is no objective measure to determine the right number of speakers or amount of speech to use in training a UBM. Empirically, from the NIST SRE we have observed no performance loss using a UBM trained with one hour of speech compared to one trained using six hours of speech. In both cases, the training speech was extracted from the same speaker population.

Adaptation of Speaker Model

In the GMM UBM system, the speaker model is derived by adapting the parameters of the UBM using the speaker's training speech and a form of Bayesian adaptation [38.28]. This is also known as Bayesian learning or *maximum a posteriori* (MAP) estimation. We use the term Bayesian adaptation since, as applied to the speaker-independent UBM to estimate the speaker-dependent model, the operation closely resembles speaker adaptation used in speech recognition applications. Unlike the standard approach of maximum-likelihood training of a model for the speaker independently of the UBM, the basic idea in the adaptation approach is to derive the speaker's model by updating the well-trained parameters in the UBM via adaptation. This provides a tighter coupling between the speaker's model and UBM, which not only produces better performance than decoupled models, but as discussed later in this section, also allows for a fast scoring technique. Like the EM algorithm, the adaption is a two-step estimation process. The first step is identical to the *expectation* step of the EM algorithm, where estimates of the sufficient statistics of the speaker's training data are computed for each mixture in the UBM. For a GMM mixture, these are the count, and the first and second moments required to compute the mixture weight, mean and variance. Unlike the second step of the EM algorithm, for adaptation these *new* sufficient statistic estimates are then combined with the *old* sufficient statistics from the UBM mixture parameters using a data-dependent mixing coefficient. The data-dependent mixing coefficient is designed so that mixtures with high counts of data from the speaker rely more on the new sufficient statistics for final parameter estimation and mixtures with low counts of data from the speaker rely more on the old sufficient statistics for final parameter estimation.

While it is possible to adapt all the parameters of the GMM during speaker model training, it has been widely shown that speaker detection performance is best when only the mean parameters are adapted [38.11, 29, 30]. Here, we will focus only on mean adaptation but details of adapting all parameters can be found in [38.30].

Given a UBM and training vectors from the hypothesized speaker, $X = \{\boldsymbol{x}_1 \ldots, \boldsymbol{x}_T\}$, we first determine the probabilistic alignment of the training vectors into the UBM mixture components (Fig. 38.2a). That is, for the mixture i in the UBM, we compute

$$\Pr(i|\boldsymbol{x}_t) = \frac{w_i p_i(\boldsymbol{x}_t)}{\sum_{j=1}^{M} w_j p_j(\boldsymbol{x}_t)} \,. \tag{38.8}$$

We then use $\Pr(i|\boldsymbol{x}_t)$ and $\boldsymbol{x}_t$ to compute the sufficient statistics for the mean parameter,

$$n_i = \sum_{t=1}^{\mathrm{T}} \Pr(i|\boldsymbol{x}_t), \tag{38.9}$$

$$E_i(\boldsymbol{x}) = \frac{1}{n_i} \sum_{t=1}^{\mathrm{T}} \Pr(i|\boldsymbol{x}_t)\boldsymbol{x}_t \,. \tag{38.10}$$

This is the same as the *expectation* step in the EM algorithm.

Lastly, these new sufficient statistics from the training data are used to update the old UBM sufficient statistics for the mixture i to create the adapted mean parameter for the mixture i (Fig. 38.2b):

$$\hat{\boldsymbol{\mu}}_i = \alpha_i E_i(\boldsymbol{x}) + (1-\alpha_i)\boldsymbol{\mu}_i \,. \tag{38.11}$$

The data-dependent adaptation coefficients controlling the balance between old and new estimates per mixture are $\{\alpha_i\}$, defined as

$$\alpha_i = \frac{n_i}{n_i + r} \,. \tag{38.12}$$

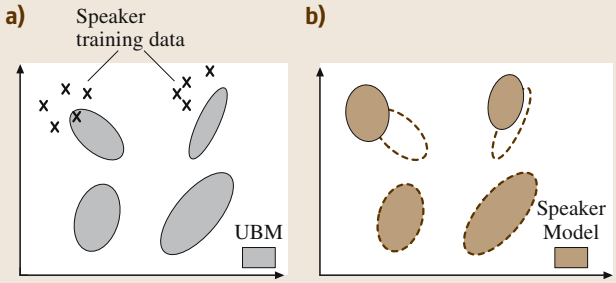


Fig. 38.2a,b Pictorial example of two steps in adapting a hypothesized speaker model. **(a)** The training vectors (*crosses*) are probabilistically mapped into the UBM mixtures. **(b)** The adapted mixture parameters are derived using the statistics of the new data and the UBM mixture parameters. The adaptation is data dependent, so UBM mixture parameters are adapted by different amounts

Here r is a fixed *relevance* factor that controls the amount of adaption from the UBM (e.g., higher values of r require more speaker data in a mixture component to adapt from the prior UBM mean).

The parameter updating as described in (38.11–38.12) can be derived from the general MAP estimation equations for a GMM using constraints on the prior distribution described in [38.28].

Using a data-dependent adaptation coefficient allows mixture-dependent adaptation of parameters. If a mixture component has a low probabilistic count, n_i, of new data, then $\alpha_i \to 0$ causing the de-emphasis of the new (potentially undertrained) parameters and the emphasis of the old (better-trained) parameters. For mixture components with high probabilistic counts, $\alpha_i \to 1$, causing the use of the new speaker-dependent parameters. The relevance factor is a way of controlling how much new data should be observed in a mixture before the new parameters begin replacing the old parameters. Values of r in the range 8–20 have been shown to work well for several speaker recognition tasks.

Comparison of published results strongly indicate that the adaptation approach provides superior performance over a decoupled system where the speaker model is trained independently of the UBM. One possible explanation for the improved performance is that the use of adapted models in the likelihood ratio is not affected by *unseen* acoustic events in recognition speech. Loosely speaking, if one considers the UBM as covering the space of speaker-independent, broad acoustic classes of speech sounds, then adaptation is the speaker-dependent *tuning* of those acoustic classes observed in the speaker's training speech. Mixture parameters for those acoustic classes not observed in the training speech are merely copied from the UBM. This means that, during recognition, data from acoustic classes unseen in the speaker's training speech produce approximately zero log-likelihood ratio values that contribute evidence neither toward nor against the hypothesized speaker. Speaker models trained using only the speaker's training speech will have low likelihood values for data from classes not observed in the training data, thus producing low likelihood ratio values. While this is appropriate for speech not from the speaker, it clearly can cause incorrect values when the unseen data occurs in test speech from the speaker.

Log-Likelihood Ratio Computation

The log-likelihood ratio for a test sequence of feature vectors X is computed as $\Lambda(X) = \log p(X|\lambda_{\text{hyp}}) - \log p(X|\lambda_{\text{ubm}})$. The fact that the hypothesized speaker model was adapted from the UBM, however, allows a faster scoring method than merely evaluating the two GMMs as in (38.6). This fast scoring approach is based on two observed effects. The first is that, when a large GMM is evaluated for a feature vector, only a few of the mixtures contribute significantly to the likelihood value. This is because the GMM represents a distribution over a large space but a single vector will be near only a few components of the GMM. Thus, likelihood values can be approximated very well using only the top-C best scoring mixture components.

The second observed effect, is that the components of the adapted GMM retain a correspondence with the mixtures of the UBM, so that vectors close to a particular mixture in the UBM will also be close to the corresponding mixture in the speaker model. Using these two effects, a fast scoring procedure operates as follows: for each feature vector, determine the top-C scoring mixtures in the UBM and compute the UBM likelihood using only these top-C mixtures. Next, score the vector against only the corresponding C components in the adapted speaker model to evaluate the speaker's likelihood. For a UBM with M mixtures, this requires only $M + C$ Gaussian computations per feature vector compared to $2M$ Gaussian computations for normal likelihood ratio evaluation. When there are multiple hypothesized speaker models for each test segment, the savings become even greater. Typical values of C are in the range 1–5.

GMM Compensation

In addition to compensation techniques applied in the feature and score domains, there are also many effec-

tive compensation techniques that can be applied in the model domain. For GMM-based classifiers, techniques that treat the undesired variability as a bias to the mean vectors have been shown to be very promising in improving the robustness of speaker recognition system. If we stack the means from a GMM into a *supervector* this can be written as

$$\boldsymbol{m}_j(s) = \boldsymbol{m}(s) + \boldsymbol{c}(s), \tag{38.13}$$

where $\boldsymbol{m}_j(s)$ is the supervector from speaker s's j-th enrollment session, $\boldsymbol{m}(s)$ is the desired compensated supervector for speaker s and $\boldsymbol{c}(s)$ is the undesired variability supervector. The main difference in the compensation techniques is in how they estimate and remove the variability vector $\boldsymbol{c}(s)$.

In speaker model synthesis (SMS) [38.31], the difference between bias vectors from a set of predefined channel types is used to synthetically generate a library of channel-dependent speaker models so as to allow matched-channel likelihood ratio scoring during recognition. Feature mapping [38.17], which is applied in the feature domain, is similar to SMS, but instead of synthetically creating unseen channel-dependent speaker models, acts to subtract off the channel-dependent biases in training and testing from the feature vectors directly.

More-recent latent factor analysis (LFA)-based techniques [38.32, 33], model the supervector bias as a low-dimensional normal bias,

$$\boldsymbol{c}(s) = U\boldsymbol{n}(s)\,, \tag{38.14}$$

where U is the low-rank session loading matrix. The LFA techniques are aimed specifically at compensation of session variability and do not require prior channel detectors or parameters. During training, the session bias is removed from each enrollment session and the session-independent supervectors are combined to produce the final speaker model. During recognition, the session bias for the test utterance is estimated and added to the speaker model to allow for matched session likelihood ratio scoring. A feature-space dual of LFA has also been developed and used successfully [38.34].

38.4.2 Support Vector Machines

A support vector machine (SVM) is a powerful classifier that has gained considerable popularity in recent years. An SVM is discriminative – it models the boundary between a speaker and a set of impostors. This approach contrasts to traditional methods for speaker recognition which separately model the probability distributions of the speaker and the general population. SVMs are based upon the structural risk minimization principal of *Vapnik* [38.35].

Two broad approaches using support vector machines have been proposed in the literature for speech applications. The first set of methods attempts to model emission probabilities for hidden Markov models [38.36–38]. This approach has been moderately successful in reducing error rates, but suffers from several problems. First, large training sets result in long training times for support vector methods. Second, the emission probabilities must be approximated [38.39], since the output of the support vector machine is not a probability. This approximation is needed to combine probabilities using the standard frame-independence assumption used in HMM speaker recognition.

A second set of methods is based upon comparing speech utterances using sequence kernels. Rather than model features from individual frames of speech, these methods instead model the entire sequence of feature vectors. Approaches include the generalized linear discriminant sequence kernel [38.40], Fisher kernel methods [38.41, 42], n-gram kernels [38.20], maximum-likelihood linear regression (MLLR) transform kernels [38.43], and GMM supervector kernels [38.44]. We focus on these latter methods, which have been successful in speaker recognition because of their accuracy and simplicity of implementation.

Basic SVM Theory

An SVM [38.35] models two classes using sums of a kernel function $K(\cdot,\cdot)$,

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i t_i K(\boldsymbol{x}, \boldsymbol{x}_i) + d, \tag{38.15}$$

where the t_i are the ideal outputs, $\sum_{i=1}^{N} \alpha_i t_i = 0$, and $\alpha_i > 0$. The vectors $\boldsymbol{x}_i$ are support vectors and obtained from the training set by an optimization process [38.45]. The ideal outputs are either 1 or -1, depending upon whether the corresponding support vector is in class 0 or class 1, respectively. For classification, a class decision is based upon whether the value, $f(\boldsymbol{x})$, is above or below a threshold.

The kernel $K(\cdot,\cdot)$ is constrained to have certain properties (the Mercer condition), so that $K(\cdot,\cdot)$ can be expressed as

$$K(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{b}(\boldsymbol{x})^{\mathrm{T}} \boldsymbol{b}(\boldsymbol{y}), \tag{38.16}$$

where $\boldsymbol{b}(\boldsymbol{x})$ is a mapping from the input space (where $\boldsymbol{x}$ lives) to a possibly infinite-dimensional *expansion*

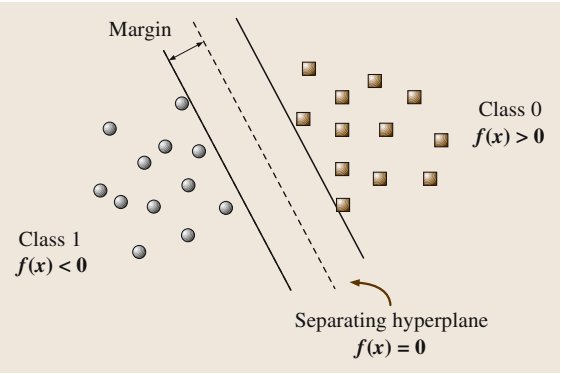


Fig. 38.3 Support vector machine concept

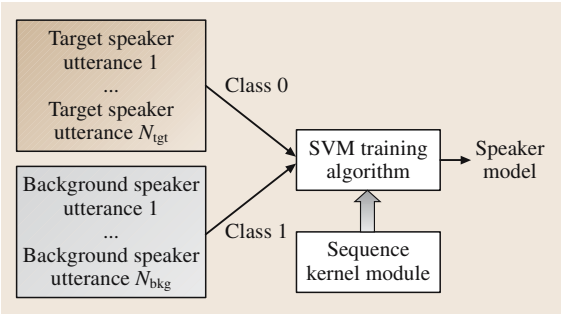


Fig. 38.4 SVM training strategy

space. The kernel is required to be positive semidefinite. The Mercer condition ensures that the margin concept is valid, and the optimization of the SVM is bounded.

The optimization condition relies upon a maximum margin concept (Fig. 38.3). For a separable data set, the system places a hyperplane in a high dimensional space so that the hyperplane has maximum margin. The data points from the training set lying on the boundaries (as indicated by solid lines in the figure) are the support vectors in (38.15). The focus, then, of the SVM training process is to model the boundary, as opposed to a traditional GMM UBM, which would model the probability distributions of the two classes.

Application of Support Vector Machines to Speaker Recognition

Discriminative training of an SVM for speaker recognition is straightforward, but several basic issues must be addressed: handling multiclass data, *world* modeling, and sequence comparison. The first two topics are handled in this section.

Since the SVM is a two-class classifier, we must recast speaker verification and identification in this structure. For speaker verification, a target model is trained for the speaker against a set of example speakers that have characteristics of the impostor population (Fig. 38.4). The set of example impostors is called a background set. In the figure, class 0 consists of all of the target speaker's utterances, and class 1 consists of the example impostors in the background set. The background speaker set is kept the same as we enroll different target speakers.

Figure 38.4 indicates the basic training strategy for SVMs using sequence kernels. Each utterance from a target or background speaker becomes a point in the SVM expansion space, see (Fig. 38.3). A sequence kernel module for comparing two utterances and producing a kernel value is implemented. The kernel module is connected into a standard SVM training tool that then produces a speaker model.

For speaker identification, a *one-versus-all* strategy is used. For a given target speaker, we pool all of the remaining nontarget speakers into class 1 and then train a speaker model. This operation is performed for all models to obtain our set of speaker identification models.

For speaker verification, the support vectors have an interesting interpretation. If $f(\boldsymbol{x})$ is an SVM for a target speaker, then we can write

$$f(\boldsymbol{x}) = \sum_{i\in\{i|t_i=1\}} \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i) - \sum_{i\in\{i|t_i=-1\}} \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i) + d. \quad (38.17)$$

The first sum can be interpreted as a per-utterance-weighted target score. The second sum has many of the characteristics of a cohort score [38.25] with some subtle differences. First, utterances rather than speakers are used as cohorts. Second, the weighting on these *cohort* utterances is not equal.

The interpretation of the SVM score as a cohort-normalized score suggests how the *world* of impostors should be modeled. Our background should have a rich speaker set, so that we can always find speakers close to the target speaker. The speakers in the background should have similar attributes – language, accent, channel type, etc. – for best performance. Note that this interpretation distinguishes the SVM approach from a universal background model method [38.30], which tries to model the impostor set with one model. Other methods for GMMs including cohort normalization [38.25] and T-norm [38.11] are closer to the SVM method; although, the latter method (T-norm) typ-

ically uses a fixed set of cohorts rather than picking out individual speakers.

Sequence Kernels for Speaker Recognition – General Structure

To apply an SVM, $f(\boldsymbol{x})$, to a speaker recognition application, we need a method for calculating kernel values from speech inputs. For recognition, a way is needed for taking a sequence of input feature vectors from an utterance, $\{\boldsymbol{x}_i\}$, and computing the SVM output, $f(\{\boldsymbol{x}_i\})$. Typically, each vector $\boldsymbol{x}_i$ would be the cepstral coefficients and deltas for a given frame of speech. One way of handling this situation is to assume that the kernel, $K(\cdot,\cdot)$, in the SVM (38.15) takes sequences as inputs, i. e., we can calculate $K(\{\boldsymbol{x}_i\},\{\boldsymbol{y}_j\})$ for two input sequences $\{\boldsymbol{x}_i\}$ and $\{\boldsymbol{y}_j\}$. This is called a sequence kernel method.

A challenge in applying the sequence kernel method is deriving a function for comparing sequences. A function is needed that, given two utterances, produces a measure of similarity of the speakers or languages. Also, a method that is efficient computationally is needed, since we will be performing many kernel inner products during training and scoring. Finally, the kernel must satisfy the Mercer condition.

One idea for constructing a sequence kernel is illustrated in Fig. 38.5. The basic approach is to compare two speech utterances, *utt 1* and *utt 2* by training a model on one utterance and then scoring the resulting model on another utterance. This process produces a value that measures the similarity between the two utterances. Two questions that follow from this approach are:

1. Can the train/test process be computed efficiently?
2. Is the resulting comparison a kernel (i. e., does it satisfy the Mercer condition)?

The answer to both of these questions is yes if the appropriate classifier is chosen – see next section.

Another idea for constructing sequence kernels is shown in Fig. 38.6. In this setup, a base model is adapted to obtain probability distributions which represent the utterances. A model comparison algorithm is then applied to get a measure of similarity. This approach has the useful property that it is naturally symmetric as long as the comparison calculation is symmetric.

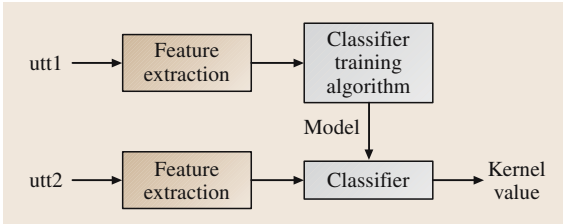


Fig. 38.5 General train/test sequence kernel

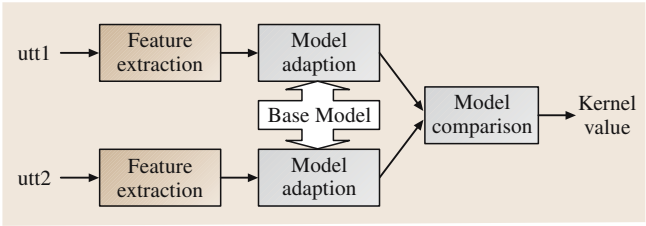


Fig. 38.6 Generative model sequence kernel

Sequence Kernels for Speaker Recognition – Specific Examples

For the train/test kernel shown in Fig. 38.5, a typical approach is the generalized linear discriminant sequence (GLDS) kernel [38.40]. In this method, the classifier is taken to be a polynomial discriminant function consisting of the monomials up to a certain degree; e.g., for two features, x_1 and x_2, and degree 2 the monomials are,

$$\boldsymbol{b}(\boldsymbol{x}) = \left(1, x_1, x_2, x_1^2, x_1x_2, x_2^2\right)^{\mathrm{T}}. \tag{38.18}$$

Suppose we have two sequences of feature vectors, $\{\boldsymbol{x}_i\}$ and $\{\boldsymbol{y}_j\}$. If a polynomial discriminant is trained using mean-squared error, then the resulting kernel is given by

$$K(\{\boldsymbol{x}_i\},\{\boldsymbol{y}_j\}) = \bar{b}_x^{\mathrm{T}}\bar{R}^{-1}\bar{b}_y. \tag{38.19}$$

In (38.19),

$$\bar{b}_x = \frac{1}{N_x}\sum_i \boldsymbol{b}(\boldsymbol{x}_i)\,, \tag{38.20}$$

i. e., $\bar{b}_x$ is the average expansion over all frames. The quantity $\bar{b}_y$ is defined similarly for the sequence, $\{\boldsymbol{y}_j\}$. The matrix, $\bar{R}$ is the correlation matrix of a background data set; typically, it is approximated with only diagonal terms. For details on the derivation of these equations, refer to [38.40].

An interesting generalization of the GLDS kernel is to replace the polynomial expansion by a general kernel using the kernel trick [38.46]. This idea leads to a method of transforming frame-level kernels into sequence kernels.

The GLDS kernel in (38.19) has the properties expected for a sequence kernel. The kernel is a single value from a sequence of vectors, symmetric, and positive definite. An interesting property of the kernel is that the kernel itself is a classifier. In a certain sense, the

SVM method acts as a method of strengthening a weaker classifier.

For the generative model sequence kernel, several methods have been proposed. Current methods are based upon adapting from a GMM or HMM base model. In [38.47], GMMs are adapted to the utterances, and an exponentiated Kullbach Leibler (KL) divergence is used to compare the utterances. In [38.43], adaptation of an HMM from a speech-to-text system is performed using maximum-likelihood linear regression (MLLR). The MLLR adaptation parameters are then compared with a weighted linear inner product. In [38.44], the adaptation is performed via MAP adaptation of a GMM. In this GMM supervector (GSV) system, the GMMs are compared using either an approximation to the KL divergence or an integral inner product. Finally [38.42] also fits loosely into this structure. A GMM model is trained from all utterance data. Then, the comparison is performed using both the base model and the target model via a modified Fisher kernel method [38.42, 48].

Nuisance Attribute Projection (NAP)

As with the GMM, compensations with SVM classifiers can also be applied directly in the model domain. The SVM nuisance attribute projection (NAP) method [38.49] works by removing subspaces that cause variability in the kernel. NAP constructs a new kernel,

$$\begin{aligned} K(\{\boldsymbol{x}_i\},\{\boldsymbol{y}_j\}) &= \left[P\bar{b}_x\right]^{\mathrm{T}}\left[P\bar{b}_y\right] \\ &= \bar{b}_x^{\mathrm{T}} P \bar{b}_y \\ &= \bar{b}_x^{\mathrm{T}}(I-\boldsymbol{v}\boldsymbol{v}^{\mathrm{T}})\bar{b}_y, \end{aligned} \tag{38.21}$$

where P is a projection ($P^2 = P$), $\boldsymbol{v}$ is the direction being removed from the SVM expansion space, $\boldsymbol{b}(\cdot)$ is the SVM expansion, and $\|\boldsymbol{v}\|_2 = 1$. The design criterion for P and correspondingly $\boldsymbol{v}$ is

$$\boldsymbol{v}^* = \underset{\boldsymbol{v},\|\boldsymbol{v}\|_2=1}{\operatorname{argmin}} \sum_{i,j} W_{i,j} \left\| P\bar{b}_{z,i} - P\bar{b}_{z,j} \right\|_2^2 , \tag{38.22}$$

where $\bar{b}_{z,i}$ is the average expansion (38.20) of the i-th sequence in a background data set. Here, $W_{i,j}$ can be selected in several different ways. If we have channel nuisance variables (e.g., electret, carbon button, cell) and a labeled background set, then we can pick $W_{i,j} = 0$ when the channels of $\bar{b}_{z,i}$ and $\bar{b}_{z,j}$ are the same, and $W_{i,j} = 1$ otherwise. Another criterion is to design the projection based on session variability. In this case, we pick $W_{i,j} = 1$ if $\bar{b}_{z,i}$ and $\bar{b}_{z,j}$ correspond to the same speaker, and $W_{i,j} = 0$ otherwise. The idea in both cases is to reduce variability in the SVM kernel distance with respect to nuisances – channel or session. See [38.44] for more details and the relationship to LFA compensation for GMMs.

Note that, while NAP is described above in the context of spectral based features, it can also be applied when using high-level token features described in the next section.

38.4.3 High-Level Feature Classifiers

In this section, we consider methods for modeling a token stream as shown in Fig. 38.7. Two standard approaches have emerged – log-likelihood ratio and support vector machine methods.

Log-Likelihood Ratio Classifier

A token sequence can be modeled using a standard *bag of n-grams* approach [38.21, 50]. For a token sequence, n-grams are produced by the standard transformation of the token stream; e.g., for bigrams, the sequence of symbols from a conversation $t_1, t_2, \ldots, t_n$ is transformed to the sequence of bigrams of symbols $t_1\_t_2, t_2\_t_3 \ldots t_{n-1}\_t_n$. Probabilities of n-grams are then found with n fixed. That is, suppose unigrams and bigrams of symbols are considered, and the unique unigrams and bigrams of the corpus are designated $d_1 \ldots d_M$ and $d_1\_d_1, \ldots, d_M\_d_M$, respectively; then calculate probabilities

$$\begin{aligned} p(d_i) &= \frac{\#(t_k = d_i)}{\sum_l \#(t_k = d_l)}, \\ p(d_i\_d_j) &= \frac{\#(t_k\_t_{k+1} = d_i\_d_j)}{\sum_{l,m} \#(t_k\_t_{k+1} = d_l\_d_m)}, \end{aligned} \tag{38.23}$$

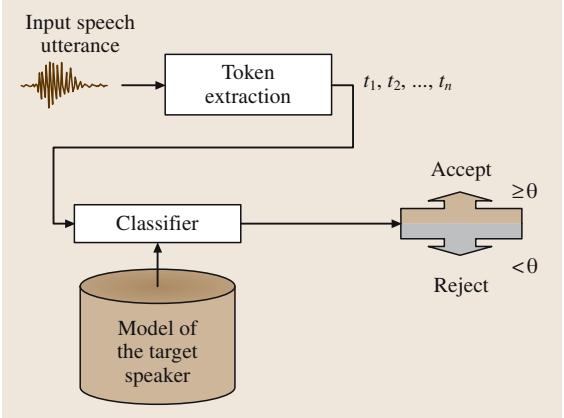


Fig. 38.7 High-level speaker recognition: basic setup

where $\#(t_k = d_i)$ indicates the number of symbols in the conversation equal to d_i, and an analogous definition is used for bigrams.

For simplicity, suppose that we have two utterances from two speakers, spk_1 and spk_2. Further suppose that the sequence of n-grams (for fixed n) in each conversation is $t_1, t_2 \ldots t_n$ and $u_1, u_2 \ldots u_m$ respectively. Denote the unique set of n-grams in the corpus as $d_1, \ldots, d_M$. A model can be built based upon the conversations for each speaker consisting of the probability of n-grams, $p(d_i|\text{spk}_j)$. The average log-likelihood ratio of the first conversation is computed as,

$$\begin{aligned}\text{score} &= \frac{1}{n}\sum_{i=1}^{n}\log\left(\frac{p(t_i|\text{spk}_2)}{p(t_i|\text{bkg})}\right)\\ &= \sum_{j=1}^{M}\frac{\#(t_i = d_j)}{n}\log\left(\frac{p(d_j|\text{spk}_2)}{p(d_j|\text{bkg})}\right)\\ &= \sum_{j=1}^{M} p(d_j|\text{spk}_1)\log\left(\frac{p(d_j|\text{spk}_2)}{p(d_j|\text{bkg})}\right), \end{aligned} \quad (38.24)$$

where $p(d_j|\text{bkg})$ is the probability of an n-gram in some large background data set. This scoring (38.24) can be easily generalized to multiple utterances per speaker; we just estimate the probabilities $p(d_j|\text{spk}_i)$ from a larger data set. Further details on this scoring method can be found in [38.21, 50].

SVM-Based Scoring

The score (38.24) can be transformed into a sequence kernel via a simple linearization; using $\log(x) \approx x - 1$ (the Taylor series expansion around $x = 1$), gives

$$\begin{aligned}\text{score} &\approx \sum_{j=1}^{M} p(d_j|\text{spk}_1)\frac{p(d_j|\text{spk}_2)}{p(d_j|\text{bkg})} - 1\\ &= \sum_{j=1}^{M}\frac{p(d_j|\text{spk}_1)}{\sqrt{p(d_j|\text{bkg})}}\frac{p(d_j|\text{spk}_2)}{\sqrt{p(d_j|\text{bkg})}} - 1. \end{aligned} \quad (38.25)$$

Once we have a sequence kernel, an SVM is trained in the same manner as the spectral-based SVMs in Fig. 38.4.

In analogy with the information retrieval literature [38.51], the result in (38.25) can be expressed in vector form. First, a vector $\boldsymbol{v}$ is constructed describing the conversation

$$\boldsymbol{v}_i = \begin{bmatrix} p(d_1|\text{spk}_i) \\ \vdots \\ p(d_M|\text{spk}_i) \end{bmatrix}. \quad (38.26)$$

In general, the vector $\boldsymbol{v}$ will be sparse since the conversation will not contain all potential unigrams, bigrams, etc. The entries of $\boldsymbol{v}$ are then weighted with a diagonal matrix, $\mathbf{D}$, with entries $D_{j,j} = 1/\sqrt{p(d_j|\text{bkg})}$. The final kernel is an inner product,

$$K(t_1^n, u_1^m) = (\mathbf{D}\boldsymbol{v}_1)^{\mathrm{T}}(\mathbf{D}\boldsymbol{v}_2)^{\mathrm{T}}. \quad (38.27)$$

The kernel (38.25) is referred to as the term frequency log-likelihood ratio (TFLLR) kernel [38.20].

The kernel in (38.25) can be generalized in a straightforward way by analyzing its components. In (38.25), the background weighting term, $1/p(d_j|\text{bkg})$, acts as a *commonality normalization*. That is, the resulting ratios in the vector (38.26) are approximately on the same scale. This normalization can be excessive if the probability estimate $p(d_j|\text{bkg})$ is small because n-gram is infrequent. To trade off between these extremes, it is natural to use a diagonal weight of the form

$$D_{j,j} = \min\left[C_j, g_j\left(\frac{1}{p(d_j|\text{bkg})}\right)\right], \quad (38.28)$$

where $g_j(\cdot)$ is a function which shrinks the dynamic range.

The general token sequence kernel obtained by combining (38.27) and (38.28) encompasses several useful weighting types. Setting $g_j(x) = \sqrt{x}$, TFLLR kernel is obtained. Second, similar to term frequency inverse document frequency (TFIDF) in information retrieval [38.51], we can use $g_j(x) = \log(x) + 1$; we refer to this weighting as a term frequency logarithmic (TFLOG) kernel. Third, an extreme case for (38.28) is to set $C_j = 1$. This results in weighting all terms equally, $D_{j,j} = 1$ for all j. Fourth, we note that both C_j and the function g_j are dependent on j. This variable weighting may make sense if we have some idea of the confidence of our estimates in the probabilities of n-grams in the background data set.

38.4.4 System Fusion

The fusion of outputs from systems modeling low- and high-level speaker information [38.19] as well as generative- and discriminative-based classifiers [38.52] have demonstrated large accuracy improvements over the constituent input systems. If the recognizers do not behave uniformly (i. e., their errors are not completely correlated) then combining the scores may make it possible to exploit complementary information that exists in the scores.

Fusion is yet another classifier that is trained on feature vectors consisting of scores from the input systems.

Ideally, a fusion classifier would be trained for each speaker, but in practice there is often limited enrollment speech per speaker that is best utilized for training the speaker model. So speaker recognition fusion systems are typically trained in a speaker-independent fashion to optimize classification of input scores as coming from the detector hypothesis H_0 (true trial) or H_1 (false trial). Vectors of system scores are computed on a development data set consisting of true and false trials from a large population of development speaker models, aggregated, and used to train a fusion classifier.

Many different fusion classifiers have been used. At the simplest level, the fusion consists of a linear combination of the system scores. The weights used in the combination can be equal, if it is assumed that each system contributes equally to overall performance and have common numeric bias and scale ranges. The weights are usually empirically adjusted on the development data set to minimize the detection error rate.

Another common and successful fusion classifier is a simple logistic regression or single-layer perceptron. This classifier is similar to the linear combination, but has a sigmoid compression function applied to the linear combination. The weights are trained using a minimum mean-squared error criteria for the target values of 0 and 1 via a form of gradient descent [38.53] on the development data. A benefit of this type of fusion classifier is that the fusion scores are often reasonable posterior probabilities estimates that can be used as confidence values.

Fusion classifiers can also use information in addition to the system scores. For example, indicators of training and/or testing data conditions, such as speech durations, spoken language or signal-to-noise measures, can be used for explicit or implicit fusion guidance. Explicit guidance is when the indicator information is used to train and apply conditional fusion classifiers, such as fusion classifiers trained on scores from speaker models trained with specific durations of enrollment speech or for speech coming from particular languages. Conditional fusion classifiers can use different system scores, for example high-level system scores depending on an English speech recognizer may be excluded when the train and/or test speech is known to not be spoken in English. Implicit guidance is when the external information is used as another element with the system scores in the fusion vector. This approach relies on the fusion classifier to learn correlations of external measures with system scores to optimize accuracy. Implicit guidance is often inferior to explicit guidance since the fusion classifiers do not have enough complexity or enough training data to learn and represent the desired conditions.

38.5 Performance Assessment

The speaker recognition approaches described in this chapter have been subject to extensive development, and this section describes their evaluation under conditions that follow protocols established by the US National Institute of Standards and Technology (NIST). Since 1995, NIST has coordinated several evaluations with the goal of assessing the existing state of the art in speaker recognition technology. The most recent speaker recognition evaluation (SRE 2006) protocol will form the basis of the performance results presented in this section. A description of the 2006 NIST speaker recognition evaluation plan can be found in [38.54].

38.5.1 Task and Corpus

The task in the NIST SRE is the basic speaker detection task described in Sect. 38.2: determine if the speaker in the training speech is the same as the speaker in the test speech. This task is evaluated for various conditions of amount of training speech, amount of test speech, recording source (telephone or auxiliary microphones), and presence of other speakers in the speech. In all there were 15 conditions in the 2006 SRE. More details of these conditions can be found in [38.55].

The data comes from the Mixer telephone speech corpus collected by the Linguistic Data Consortium (LDC) [38.56] which consists of thousands of telephone conversations between hundreds of speakers within the US. The speakers cover a distribution of age, gender, location, and native languages. The participants in a telephone call are given general topics to discuss, but the conversations are unscripted and about five minutes in duration. Participants were encouraged to make many calls to the system over several weeks and to use varied telephone instruments and locations to provide large session and handset variability in the data. Participants

that have a common native non-English language are also instructed to conduct conversations in their native language.

For results presented here we focus on the one and eight conversation-side train and one conversation-side test telephone speech conditions. Each conversation-side provides nominally 2.5 minutes of speech. To induce mismatch, training data for a speaker comes from a single telephone number and the test speech comes from a different telephone number. Results from other conditions can be found in [38.55, 57].

Performance is shown in terms of equal error rate (EER), minimum decision cost function (minDCF) [38.54] and detection error tradeoff (DET) curves [38.58]. The minDCF is defined as

$$\text{minDCF} = \min_{\theta} 0.1\,P_{\text{miss}}(\theta) + 0.99\,P_{\text{fa}}(\theta), \qquad (38.29)$$

where $P_{\text{miss}}(\theta)$ and $P_{\text{fa}}(\theta)$ are the probability of miss and false alarm, respectively, at the detection threshold θ.

38.5.2 Systems

Below are some speaker detection systems evaluated on the 2006 SRE that exemplify the features and classifiers detailed in this chapter. More details and other system used in the 2006 SRE can be found in [38.59].

GMM UBM
Features. Mel cepstra plus delta cepstra
Classifier. 2048 mixture GMM UBM
Compensations. RASTA and mean/variance normalization in feature domain, LFA in model domain, Z-norm, and T-norm in score domain

SVM-GLDS
Features. Mel cepstra and LPC cepstra plus delta cepstra
Classifier. SVM with GLDS kernel
Compensations. RASTA and mean/variance normalization in feature domain, NAP in model domain, T-norm in score domain

SVM-GSV
Features. Mel cepstra plus delta cepstra
Classifier. SVM with generative model sequence kernel using GMM UBM mean supervectors
Compensations. NAP in model domain, T-norm in score domain

LLR-WORD
Features. Speech recognition system word sequences from word lattices
Classifier. n-gram log-likelihood ratio
Compensations. Z-norm and T-norm in score domain

SVM-WORD
Features. Speech recognition system word sequences from word lattices
Classifier. SVM TFLOG kernel using n-gram vectors
Compensations. None

38.5.3 Results

In Fig. 38.8, we show performance for the one conversation train and one conversation test condition in terms of minDCF and EER for the above individual systems, fusion of all systems and fusion of the spectral-based systems only (GMM-UBM, SVM-GLDS, and SVM-GSV). Figure 38.9 shows the same for the eight conversation train and one conversation test condition. Results for both figures use all trials from the train/test condition (e.g., both English and non-English conversations).

These results demonstrate some general observations about the various speaker recognition systems. First, we see that the spectral based systems have comparable performance to each other and significantly better performance than systems using word-based tokens. We also see that increased training data improves performance for all systems. For word-token-based systems this gain is attributable to increased instances of sparse speaker-dependent idiolect events. The improvement in

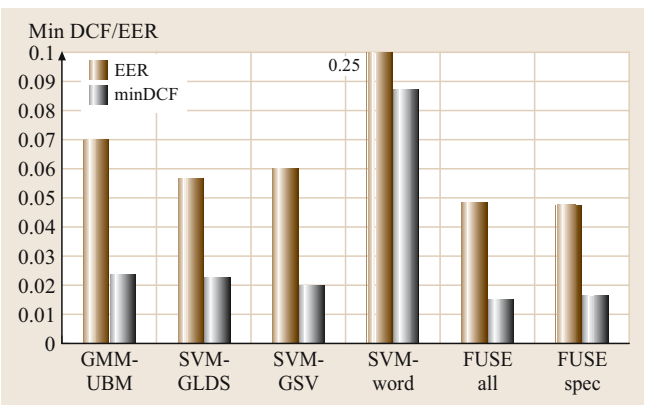


Fig. 38.8 Performance in terms of minDCF and EER for one conversation train and one conversation test for low-level, high-level, and fusion systems

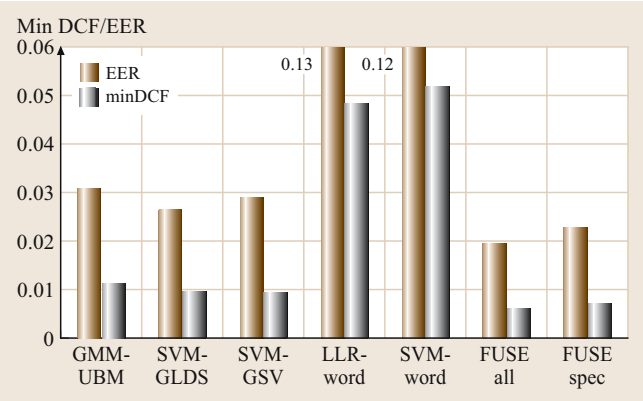


Fig. 38.9 Performance in terms of minDCF and EER for eight conversation trains and one conversation test for low-level, high-level, and fusion systems

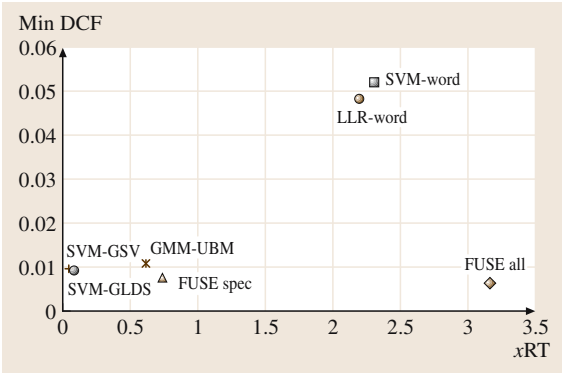


Fig. 38.10 MinDCF versus real-time factor for eight conversation trains and one conversation test for low-level, high-level, and fusion systems

spectral based systems is primarily due to observing more session variability of the speaker's speech. Lastly we observe that system fusion provides further gains in performance. This gain is seen when fusing only spectral based systems as well as when fusing spectral and word-token-based system. The extra gain from fusing in the word-token-based systems is virtually zero for the one conversation train condition and about 16% relative at EER for the eight conversation train condition.

While not shown, the results when restricting the trials to English-only conversations are only slightly better than using all trials. On average the EER is decreased by about 0.75%. An analysis of cross-lingual train/test condition indicates that spectral-based speaker recognition systems are relatively robust to the spoken language. High-level system that depend on matching the input language to the language of the speech recognizer are much more susceptible to degradations under cross-lingual conditions.

38.5.4 Computational Considerations

Although the focus of the discussion of speaker recognition systems has been performance, any practical implementation of such a system needs to account for computational complexity. A tradeoff plot of minDCF versus real-time processing factor for the above systems is shown in Fig. 38.10. The real-time processing factor is the multiple of input speech duration a system requires to produce a speaker detection score. (e.g., 2xRT means a 1 minute of speech requires 2 minutes of processing time. Real-time factors were computed using common input and CPU.) As exemplified here, since high-level systems often rely on the output of a speech recognition system, there is a significant computational cost for the incremental gains in fusing high-level systems with low-level systems. However, in applications where a speech recognition system is already being used for other purposes, reusing the output for high-level speaker recognition systems may be practical.

38.6 Summary

This chapter has presented a variety of methods for implementing automatic text-independent speaker verification systems. The speaker verification problem was cast as a general likelihood ratio detection task where features conveying speaker information from low-level spectral information to high-level word sequences are extracted and compared to models of speaker-dependent and general speaker-independent characteristics. Two of the most widely used classifiers, the generative GMM-UBM and the discriminative SVM, were presented as methods for performing the model generation and likelihood ratio comparison. To handle degradations due to channel and session variability compensation techniques in the feature, model, and score domains were discussed. Finally results from the 2006 NIST speaker recognition evaluation showed the relative performance of systems built using low- and high-level features with generative and discriminative classifiers and well as

fusions of these systems. Overall performance trends indicate that text-independent speaker verification on conversational telephone speech has shown significant improvement over years of research with state-of-the-art systems producing EERs below 2%. For the latest advances in the field of automatic language recognition, the reader is referred to the most recent proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), the Odyssey Speaker and Language Recognition Workshop, and the International Conference on Spoken Language Processing (Interspeech – ICSLP).

References

38.1 J. Naik, G. Doddington: Evaluation of a high performance speaker verification system for access Control, Proc. ICASSP (1987) pp. 2392–2395

38.2 A. Higgins, L. Bahler, J. Porter: Speaker verification using randomized phrase prompting, Digital Signal Process. **1**, 89–106 (1991)

38.3 J. Naik, L. Netsch, G. Doddington: Speaker verification over long distance telephone lines, Proc. ICASSP (1989) pp. 524–527

38.4 C. Schmandt, B. Arons: A conversational telephone messaging system, IEEE Trans. Consumer Electron. **30**(3), xxi–xxiv (1984)

38.5 L. Wilcox, F. Chen, D. Kimber, V. Balasubramanian: Segmentation of speech using speaker identification, Proc. ICASSP (1994) pp. I.161–I.164

38.6 B.M. Arons: *Interactively Skimming Recorded Speech*, Ph.D. Thesis (MIT Press, Cambridge 1994)

38.7 G. Doddington: Speaker recognition – identifying people by their voices, Proc. IEEE **73**(11), 1651–1664 (1985)

38.8 R.B. Dunn, D.A. Reynolds, T.F. Quatieri: Approaches to speaker detection and tracking in multi-speaker audio, Digital Signal Process. **10**(1), 93–112 (2000)

38.9 K. Li, J. Porter: Normalizations and selection of speech segments for speaker recognition scoring, Proc. ICASSP (1988) pp. 595–598

38.10 D.A. Reynolds, T.F. Quatieri, R.B. Dunn: Speaker verification using adapted Gaussian mixture models, Digital Signal Process. **10**(1), 19–41 (2000)

38.11 R. Auckenthaler, M. Carey, H. Lloyd-Thomas: Score Normalization for Text-Independent Speaker Verification Systems, Digital Signal Process. **10**, 42–54 (2000)

38.12 W.M. Campbell, D.A. Reynolds, J.P. Campbell, K. Brady: Estimating and evaluating confidence for forensic speaker recognition, Proc. ICASSP (2005) pp. 18–23

38.13 D. Reynolds, W. Andrews, J. Campbell, J. Navratil, B. Peskin, A. Adami, Q. Jin, D. Klusacek, J. Abramson, R. Mihaescu, J. Godfrey, D. Jones, B. Xiang: The SuperSID project: Exploiting high-level information for high-accuracy speaker recognition, Proc. ICASSP (2003) pp. 784–787

38.14 F. Bimbot, J.-F. Bonastre, C. Fredouille, G. Gravier, I. Magrin-Chagnolleau, S. Meignier, T. Merlin, J. Ortega-Garcia, D. Petrovska-Delacretaz, D.A. Reynolds: A tutorial on text-independent speaker verification, EURASIP J. Appl. Signal Process. **4**, 430–451 (2004)

38.15 F. Soong, A. Rosenberg: On the use of instantaneous and transitional spectral information in speaker recognition, Proc. ICASSP (1986) pp. 877–880

38.16 H. Hermansky, N. Morgan, A. Bayya, P. Kohn: RASTA-PLP speech analysis technique, Proc. ICASSP (1992) pp. I.121–I.124

38.17 D.A. Reynolds: Channel robust speaker verification via feature mapping, Proc. ICASSP (2003) pp. I.53–I.56

38.18 L. Nguyen, S. Matsoukas, J. Davenport, F. Kubala, R. Schwartz, J. Makhoul: Progress in transcription of broadcast news using Byblos, Speech Commun. **38**(1-2), 213–230 (2002)

38.19 J.P. Campbell, D.A. Reynolds, R.B. Dunn: Fusing high- and low-level features for speaker recognition, Proc. European Conf. Speech Commun. Technol. (2003) pp. 2665–2668

38.20 W.M. Campbell, J.P. Campbell, D.A. Reynolds, D.A. Jones, T.R. Leek: High-level speaker verification with support vector machines, Proc. ICASSP (2004) pp. I.73–I.76

38.21 G. Doddington: Speaker recognition based on idiolectal differences between speakers, Proc. European Conf. Speech Commun. Technol. (2001)

38.22 A. Dempster, N. Laird, D. Rubin: Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. **39**, 1–38 (1977)

38.23 D.A. Reynolds: Speaker identification and verification using gaussian mixture speaker models, Speech Commun. **17**(1-2), 91–108 (1995)

38.24 D.A. Reynolds: Automatic speaker recognition using Gaussian mixture speaker models, Lincoln Lab. J. **8**(2), 173–192 (1995)

38.25 A.E. Rosenberg, J. DeLong, C.H. Lee, B.H. Juang, F.K. Soong: The use of cohort normalized scores for speaker verification, Int. Conf. Speech Lang. Process. (1992) pp. 599–602

38.26 M. Carey, E. Parris, J. Bridle: A speaker verification system using alphanets, Proc. ICASSP (1991) pp. 397–400

38.27 D.A. Reynolds: Comparison of background normalization methods for text-independent speaker verification, Proc. European Conf. Speech Commun. Technol. (1997) pp. 963–967

38.28 J.L. Gauvain, C.-H. Lee: Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains, IEEE Trans. Speech Audio Process. **2**(2), 291–298 (1994)

38.29 S. Vuuren: *Speaker Verification in a Time-Feature Space*, Ph.D. Thesis (OGI, Beaverton 1999)

38.30 D.A. Reynolds, T.F. Quatieri, R. Dunn: Speaker verification using adapted Gaussian mixture models, Digital Signal Process. **10**(1–3), 19–41 (2000)

38.31 R. Teunen, B. Shahshahani, L. Heck: A model-based transformational approach to robust speaker recognition, Proc. Int. Conf. Spoken Lang. Process. (2000)

38.32 P. Kenny, G. Boulianne, P. Dumouchel: Eigenvoice modeling with sparse training data, IEEE Trans. Speech Audio Process. **13**(3), 345–354 (2005)

38.33 R. Vogt, B. Baker, S. Sriharan: Modelling session variability in text-independent speaker verification, Proc. Interspeech (2005) pp. 3117–3120

38.34 C. Vair, D. Colibro, F. Castaldo, E. Dalmasso, P. Laface: Channel factors compensation in model and feature domain for speaker recognition, Proc. Odyssey Speaker and Language Workshop (2006)

38.35 N. Cristianini, J. Shawe-Taylor: *Support Vector Machines* (Cambridge Univ. Press, Cambridge 2000)

38.36 M. Schmidt, H. Gish: Speaker identification and verification using Gaussian mixture speaker models, Proc. ICASSP (1996) pp. 105–108

38.37 V. Wan, W.M. Campbell: Support vector machines for verification and identification, neural networks for signal processing X, Proc. 2000 IEEE Signal Process. Workshop (2000) pp. 775–784

38.38 A. Ganapathiraju, J. Picone: Hybrid SVM/HMM architectures for speech recognition, Speech Transcription Workshop (2000)

38.39 J.C. Platt: Probabilities for SV machines. In: *Advances in Large Margin Classifiers*, ed. by A.J. Smola, P.L. Bartlett, B. Schölkopf, D. Schuurmans (MIT Press, Cambridge 2000) pp. 61–74

38.40 W.M. Campbell: Generalized linear discriminant sequence kernels for speaker recognition, Proc. ICASSP (2002) pp. 161–164

38.41 S. Fine, J. Navrátil, R.A. Gopinath: A hybrid GMM/SVM approach to speaker recognition, Proc. ICASSP (2001)

38.42 V. Wan, S. Renals: SVMSVM: support vector machine speaker verification methodology, Proc. ICASSP (2003) pp. 221–224

38.43 A. Stolcke, L. Ferrer, S. Kajarekar, E. Shriberg, A. Venkataraman: MLLR transforms as features in speaker recognition, Proc. European Conf. Speech Commun. Technol. (2005) pp. 2425–2428

38.44 W.M. Campbell, D.E. Sturim, D.A. Reynolds: Support vector machines using GMM supervectors for speaker verification, IEEE Sign. Process. Lett. **13**(5), 308–311 (2006)

38.45 R. Collobert, S. Bengio: SVMTorch: Support vector machines for large-scale regression problems, J. Machine Learn. Res. **1**, 143–160 (2001)

38.46 J. Louradour, K. Daoudi, F. Bach: SVM speaker verification using an incomplete Cholesky decomposition sequence kernel, IEEE 2006 Odyssey: The Speaker and Language Recognition Workshop (2006)

38.47 P.J. Moreno, P. Ho, N. Vasconcelos: A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. In: *Advances in Neural Information Processing 16*, ed. by S. Thrun, L.K. Saul, B. Schölkopf (MIT Press, Cambridge 2004)

38.48 T.S. Jaakkola, D. Haussler: Exploiting generative models in discriminative classifiers. In: *Advances in Neural Information Processing 11*, ed. by M.S. Kearns, S.A. Solla, D.A. Cohn (MIT Press, Cambridge 1998) pp. 487–493

38.49 A. Solomonoff, W.M. Campbell, I. Boardman: Advances in channel compensation for SVM speaker recognition, Proc. ICASSP (2005)

38.50 W.D. Andrews, M.A. Kohler, J.P. Campbell, J.J. Godfrey, J. Hernández-Cordero: Gender-dependent phonetic refraction for speaker recognition, Proc. ICASSP (2002) pp. I.149–I.153

38.51 T. Joachims: *Learning to Classify Text Using Support Vector Machines* (Kluwer Academic, Dordrecht 2002)

38.52 W.M. Campbell, D.A. Reynolds, J.P. Campbell: Fusing discriminative and generative methods for speaker recogntion: experiments on Switchboard and NFI/TNO field data, Proc. Odyssey Speaker and Language Workshop (2004) pp. 41–44

38.53 L. Kukolich, R. Lippman: *LNKnet User's Guide* Manual and software available online at http://www.ll.mit.edu/IST/lnknet (2004)

38.54 *The 2006 NIST Speaker Recognition Evaluation Plan*http://www.nist.gov/speech/tests/spk/2006/sre-06_evalplan-v9.pdf (2006)

38.55 M.A. Pryzbocki, A.F. Martin, A.N. Le: NIST speaker recognition evaluation chronicles part 2, Proc. Odyssey Speaker and Language Workshop (2006)

38.56 J.P. Campbell, H. Nakasone, C. Cieri, D. Miller, K. Walker, A.F. Martin, M.A. Pryzbocki: The MMSR bilingual and crosschannel corpora for speaker recognition research and evaluation, Proc. Odyssey Speaker and Language Workshop (2004) pp. 29–32

38.57 D.E. Sturim, W.M. Campbell, D.A. Reynolds, R.B. Dunn, T.F. Quatieri: Robust speaker recognition with cross-channel data: MIT-LL results on the 2006 NIST SRE auxiliary microphone task, Proc. ICASSP (2007)

38.58 A. Martin, G. Doddington, T. Kamm, M. Ordowski, M. Pryzbocki: The DET curve in assessment of detection task performance, Proc. European Conf. Speech Commun. Technol. (1997) pp. 1895–1898

38.59 W.M. Campbell, D.E. Sturim, W. Shen, D.A. Reynolds, J. Navratil: The MIT-LL/IBM 2006 speaker recognition system: High-performance reduced-complexity recognition, Proc. ICASSP (2007)

41. Automatic Language Recognition Via Spectral and Token Based Approaches

D. A. Reynolds, W. M. Campbell, W. Shen, E. Singer

Automatic language recognition from speech consists of algorithms and techniques that model and classify the language being spoken. Current state-of-the-art language recognition systems fall into two broad categories: spectral- and token-sequence-based approaches. In this chapter, we describe algorithms for extracting features and models representing these types of language cues and systems for making recognition decisions using one or more of these language cues. A performance assessment of these systems is also provided, in terms of both accuracy and computation considerations, using the National Institute of Science and Technology (NIST) language recognition evaluation benchmarks.

41.1 Automatic Language Recognition

Automatic language recognition from speech consists of algorithms and techniques that model and classify the language being spoken. The term recognition can refer to either identification of the spoken language from a set of languages known to the system, or detection of the presence of a particular language out of a larger set of languages unknown to the system. As with other information conveyed by the speech signal, cues about the language being spoken occur at several levels. At the top level the words being spoken to communicate a coherent message are the basis of what defines a language. However, relying on automatic extraction of the words presupposes the availability of a reliable speech recognizer in the languages of interest, which is often not the case in many practical applications requiring language recognition. Further, language recognition is often used as a low computation pre-processor to determine which higher computation speech recognizer should be run, if any. Fortunately, cues about the language are also transmitted at lower levels in the speech signal, such as in the phoneme inventory and co-occurrences (phonotactics), the rhythm and timing (prosodics) and the acoustic sounds (spectral characteristics), that are more amenable to automatic extraction and modeling.

Current state-of-the-art language recognition systems are based on processing these low-level cues and fall into two broad categories: spectral based and token sequence based. This chapter describes algorithms for extracting features and models representing these types of language cues and systems for making recognition decisions using one or more of these language cues. Additionally, a performance assessment of these systems is provided, in terms of both accuracy and computation considerations, using the National Institute of Science and Technology (NIST) language recognition evaluation benchmarks.

41.2 Spectral Based Methods

Spectral-based methods for language recognition operate by extracting measurements of the short-term speech spectrum over fixed analysis frames and then modeling characteristics of these measurements, or features, for each language to be recognized. Classification techniques that have proved successful for language recognition are generative, via Gaussian mixture models (GMMs), and discriminative, via support vector machines (SVMs). This section first describes the spectral feature extraction process and then the GMM and SVM classifiers used for recognition.

41.2.1 Shifted Delta Cepstral Features

The features used for language recognition systems are based on the cepstral coefficients derived from a mel-scale filterbank analysis typically used in other speech processing tasks such as speech and speaker recognition. A block diagram of the filterbank feature extraction system is shown in Fig. 41.1. The feature extraction consists of the following steps. Every 10 ms the speech signal is multiplied by a Hamming window with a duration of 20 ms to produce a short-time speech segment for analysis. The discrete Fourier spectrum is obtained via a fast Fourier transform (FFT) from which the magnitude squared spectrum is computed. The magnitude spectrum is multiplied by a pre-emphasis filter to emphasize the high-frequency portion of the spectrum and the result is put through a bank of triangular filters. The filterbank used is similar to that in [41.1] and simulates critical band filtering with a set of triangular bandpass filters that operate directly on the magnitude spectrum. The critical band warping is done by approximating the mel-frequency scale, which is linear up to 1000 Hz and logarithmic above 1000 Hz. The center frequencies of the triangular filters follow a uniform 100 Hz mel-scale spacing, and the bandwidths are set so the lower and upper passband frequencies of a filter lie on the center frequencies of the adjacent filters, giving equal bandwidths on the mel-scale but increasing bandwidths on the linear frequency scale. The number of filters is selected to cover the signal bandwidth $[0, f_s/2]$ Hz, where f_s is the sampling frequency. For 8 kHz sampled telephone speech, there are 24 filters.

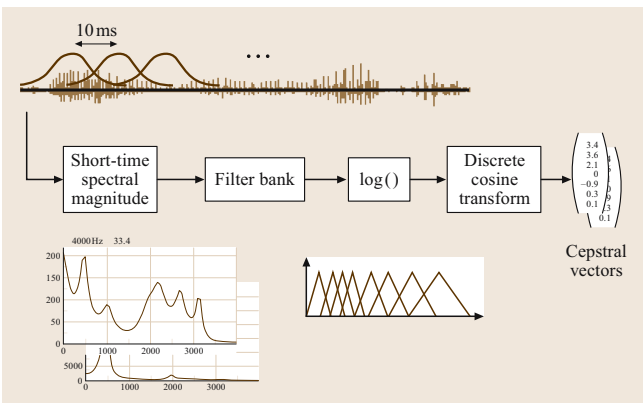


Fig. 41.1 Mel-scale filterbank cepstral feature extraction

The log energy output of each filter is then used as an element of a filterbank feature vector and the short-term mel-scale cepstral feature vector is obtained from the inverse cosine transform of the filterbank vector. To model short-term speech dynamics, the static cepstral vector is augmented by difference (delta) and acceleration (delta–delta) cepstra computed across several frames.

Since the speech used for training and testing language recognition systems can come from a variety of sources (involving different microphones and channels), it is important to apply some form of channel compensation to the features. Typical channel compensation techniques include blind deconvolution via RASTA (relative spectral) filtering [41.2] and per-utterance feature normalization to zero mean and unit variance. More-sophisticated compensation techniques, such as feature mapping [41.3], that explicitly model channel effects are also successfully used.

One of the significant advances in performing language recognition using GMMs was the discovery of a better feature set for language recognition [41.4]. The improved feature set, the shifted delta cepstral (SDC) coefficients, are an extension of delta-cepstral coefficients.

SDC coefficients are calculated as shown in Fig. 41.2. SDC coefficients are specified by four parameters, conventionally written as N-d-P-k. For a frame of data at time t, a set of MFCCs are calculated, i. e.,

$$c_0(t), c_1(t), \ldots, c_{N-1}(t) \,. \tag{41.1}$$

Note that the coefficient $c_0(t)$ is used. The parameter d determines the spread across which deltas are calculated, and the parameter P determines the gaps between successive delta computations. For a given time t,

$$\Delta c(t, i) = c(t + iP + d) - c(t + iP - d) \tag{41.2}$$

represents an intermediate calculation, where $i = 0, 1, \ldots, k$. The SDC coefficients are then k stacked

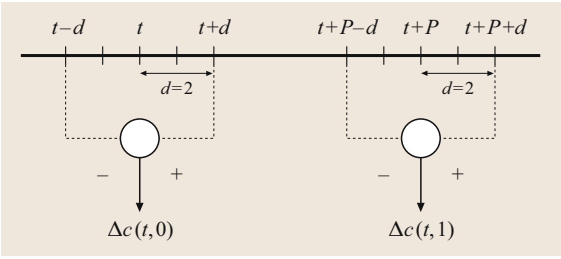


Fig. 41.2 Shifted delta cepstral coefficients

versions of (41.2),

$$\mathrm{SDC}(t) = \begin{pmatrix} \Delta c(t,0) \\ \Delta c(t,1) \\ \vdots \\ \Delta c(t,k-1) \end{pmatrix} . \tag{41.3}$$

Prior to the use of SDC coefficients, GMM-based language recognition was less accurate than alternate approaches [41.5]. SDC coefficients capture variation over many frames of data; e.g., the systems described in this chapter use 21 consecutive frames of cepstral coefficients. This long-term analysis might explain the effectiveness of the SDC features in capturing language-specific information.

41.2.2 Classifiers

Current state-of-the-art language recognition systems use either or both generative GMM-based classifiers and discriminative SVM-based classifiers.

Gaussian Mixture Models

The approach for a GMM classifier is to model the distribution of cepstral features for each language to be recognized. A language-specific GMM is given by

$$p(\boldsymbol{x}|\lambda) = \sum_{i=1}^{M} \pi_i \mathcal{N}_i(\boldsymbol{x}) , \tag{41.4}$$

where $\boldsymbol{x}$ is a D-dimensional feature vector, $\mathcal{N}_i(\boldsymbol{x})$ are the component densities, and π_i are the mixture weights. Each component density is a D-variate Gaussian function of the form

$$\mathcal{N}_i(\boldsymbol{x}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}_i|^{1/2}} \times \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^{\mathrm{T}}\, \boldsymbol{\Sigma}_i^{-1}\, (\boldsymbol{x}-\boldsymbol{\mu}_i)\right\} , \tag{41.5}$$

with mean vector $\boldsymbol{\mu}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$. The mixture weights satisfy the constraint

$$\sum_{i=1}^{M} \pi_i = 1 , \tag{41.6}$$

which ensures that the mixture is a true probability density function. The complete language-specific Gaussian mixture density is parameterized by the mean vectors, covariance matrices, and mixture weights from all component densities and is represented by the notation

$$\lambda = \{\pi_i,\ \boldsymbol{\mu}_i,\ \boldsymbol{\Sigma}_i\} \quad i = 1, \ldots, M . \tag{41.7}$$

For language recognition using SDC features, diagonal covariances and a mixture order of $M = 2048$ are used. It was empirically determined [41.4] that higher-order GMMs provided substantial performance improvements when using SDC features.

The GMM parameters are estimated via the expectation-maximization (EM) algorithm [41.6] or by Bayesian adaptation from a universal background model (UBM) [41.7], as is done in speaker recognition (see Part F of this Handbook). Adapting from a background model has computational advantages in that it speeds up both model training and likelihood computations. For language recognition, the UBM is typically constructed by training a GMM using an aggregation of the training data from all available languages. Recently, discriminative training of GMM parameters based on maximum mutual information optimization has yielded very promising results [41.8].

For a general language recognition task, a GMM, λ_l, is trained for each of the desired languages, $l = 1, \ldots, L$, and the likelihood of a sequence of feature vectors, $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T)$, extracted from a test utterance is computed as

$$p(\boldsymbol{X}|\lambda_l) = \prod_{t=1}^{T} p(\boldsymbol{x}_t|\lambda_l) . \tag{41.8}$$

The model likelihood is computed assuming independence between the feature vectors which means the temporal ordering of the feature vectors is unimportant. However, the SDC feature vectors $\boldsymbol{x}_t$ were themselves extracted over a multiframe time span and thereby encode local temporal sequence information.

The likelihood scores from the set of language models are then used by the back-end fusion/decision system to compute likelihood ratios or to fuse with other system scores (Sect. 41.4).

Support Vector Machines

Support vector machines (SVMs) are flexible classifiers that have recently shown promise in language recognition. SVMs rely on separating data in high-dimensional spaces using the maximum margin concept [41.9]. An SVM, $f(\boldsymbol{Z})$, is constructed from sums of a kernel function $K(\cdot,\cdot)$,

$$f(\boldsymbol{Z}) = \sum_i \alpha_i K(\boldsymbol{Z}, \boldsymbol{Z}_i) + d \,, \tag{41.9}$$

where $\boldsymbol{Z}$ is the input feature vector, $\sum_i \alpha_i = 0$, and $\alpha_i \neq 0$. The $\boldsymbol{Z}_i$ are support vectors and are obtained from the training set by an optimization process [41.10]. The ideal classifier values are either 1 or -1, depending on whether the corresponding support vector is in class 0 or class 1. For classification, a class decision is based on whether the value $f(\boldsymbol{Z})$ is above or below a threshold.

SVM kernels provide a method for comparing sequences of feature vectors (e.g., shifted delta cepstral coefficients). Given sequences of feature vectors from two utterances, $\boldsymbol{X} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{T_x})$ and $\boldsymbol{Y} = (\boldsymbol{y}_1, \cdots, \boldsymbol{y}_{T_y})$, the kernel produces a comparison that provides discrimination between the languages of the utterances. If the utterances are from the same language, a large positive value is desired; otherwise, the kernel should produce a large negative value.

The general setup for training a language recognition system using support vector machines is a *one-versus-all* strategy as shown in Fig. 41.3. For the example of English in the figure, class 1 contains only target (English) language data and class 0 contains all of the nontarget (non-English) language data pooled together. Training proceeds using a standard SVM training tool with a sequence kernel module (e.g., [41.10]). The resulting process produces a model for recognizing the target language.

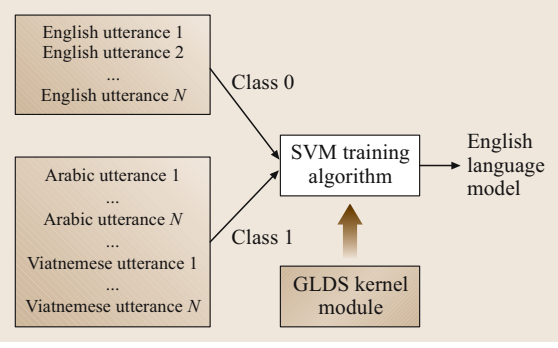


Fig. 41.3 Training a support vector machine for language recognition

A useful kernel for language recognition is the generalized linear discriminant sequence (GLDS) kernel [41.11]. The GLDS kernel is simply an inner product between average high-dimension expansions of a set of basis functions $b_j(\cdot)$,

$$\boldsymbol{b}(\boldsymbol{x}) = (b_1(\boldsymbol{x}) \;\cdots\; b_K(\boldsymbol{x}))^{\mathrm{T}} \,, \tag{41.10}$$

where K is the number of basis functions. Typically, monomials up to a given degree are used as basis functions. An example basis is

$$\boldsymbol{b}(\boldsymbol{x}) = \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_n \\ x_1^2 \\ \vdots \\ x_1^{i_1} x_2^{i_2} \ldots x_n^{i_n} \\ \vdots \end{pmatrix} . \tag{41.11}$$

The formula for the kernel is

$$K_{\mathrm{GLDS}}(\boldsymbol{X}, \boldsymbol{Y}) = \bar{\boldsymbol{b}}_x^{\mathrm{T}} \bar{\boldsymbol{R}}^{-1} \bar{\boldsymbol{b}}_y \,. \tag{41.12}$$

The mapping $\boldsymbol{X} \to \bar{\boldsymbol{b}}_x$ is defined as

$$\boldsymbol{X} \to \frac{1}{T_x} \sum_{t=1}^{T_x} \boldsymbol{b}(\boldsymbol{x}_t) \,. \tag{41.13}$$

The vector $\bar{\boldsymbol{b}}_y$ is defined in an analogous manner to (41.13). $\bar{\boldsymbol{R}}$ is a correlation matrix derived from a large set of data from multiple languages and many speakers and is usually diagonal.

In a general language recognition task, a set of SVM language models $\{f_l\}$ are trained to represent the target languages $l = 1, \cdots, L$. The sequence of vectors $\boldsymbol{X}$ is extracted from a test utterance, mapped to an average expansion using (41.13), and converted to a set of scores $s_l = f_l(\bar{\boldsymbol{b}}_x)$. The SVM scores s_l are usually transformed to s'_l using a likelihood-ratio-type calculation

$$s'_l = s_l - \log\left(\frac{1}{M-1} \sum_{j \neq l} e^{-s_j}\right) . \tag{41.14}$$

The transformation (41.14) assumes that SVM scores behave like log-likelihood ratios (see [41.12, 13] for related discussions).

Several computational simplifications are available for implementing an SVM system for language recognition. For instance, language recognition scoring can be reduced to a single inner product. Also, language model training can be simplified to a linear kernel in Fig. 41.3. The reader is referred to [41.13] for additional details.

41.3 Token-Based Methods

Unlike acoustic approaches in which fixed-duration windows of the speech signal are used to extract features for classification, token-based approaches segment the input speech signal into logical units or tokens. These units could be based on a phonetic segmentation (e.g., phones), or a data-driven segmentation (e.g., GMM mixture component tokens [41.14]). Features are then extracted from the token stream, and classification is performed using a language model. Figure 41.4 illustrates the process.

This basic architecture allows for a variety of token-based approaches, but the phoneme recognition followed by language modeling (PRLM) approach originally described in [41.15, 16] has been shown to work well for language recognition. The remainder of this section will focus on PRLM and related techniques that model the phonotactic properties (phone stream dependencies) of languages. Other token-based techniques that should be noted include the work of [41.14] in which abstract tokens were used in place of phonetic units to achieve near state-of-the-art language recognition performance.

41.3.1 Tokens

In the PRLM framework, the tokenizer is a phone recognizer (PR) that converts an input speech signal into a sequence of phonetic units (phones). Each phone and its surrounding contexts are then combined to form phone n-grams, the features used for scoring. These features are scored against n-gram language models that correspond to specific target languages to make language decisions. Intuitively, these features represent the phonotactic dependencies (the rules governing the distribution of phonetic sounds within a language) between individual phonetic units. In an early work [41.17], *House* and *Neuberg* used transcribed data to demonstrate the feasibility of exploiting phonotactics to perform automatic language identification. Figure 41.5 illustrates the process.

Phone Recognition

Phone recognition is typically accomplished using hidden Markov models (HMMs) to represent phone units. Figure 41.6 shows a standard topology for different phonetic units. During recognition, these models are combined to allow any phone to transition to any other phone with equal probability. This configuration is often referred to as an *open phone loop* or *null* grammar, as shown in Fig. 41.7.

The phone decoding process is based on a sequence of observations $\boldsymbol{X} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T)$, where T is the number of speech frames. The observation $\boldsymbol{x}_t$ is a vector of acoustic features, usually mel-frequency cepstrum coefficients (MFCC) [41.1] or perceptual linear predic-

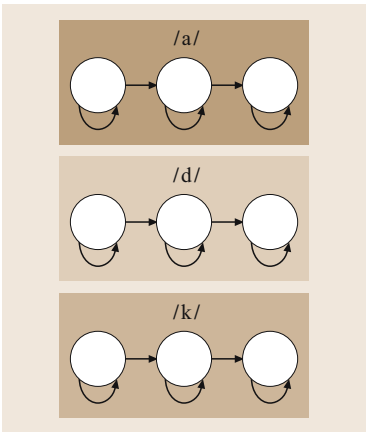


Fig. 41.6 HMM: typical HMM topology for phonetic units

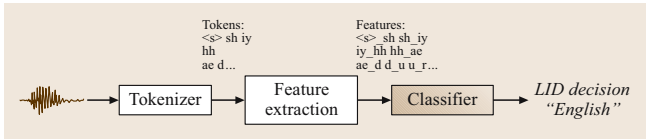


Fig. 41.4 Token-based language recognition

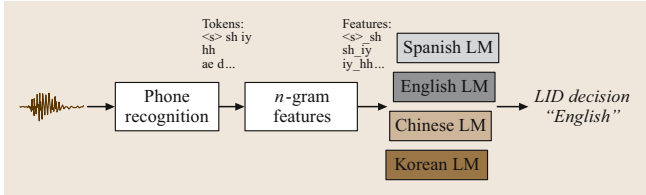


Fig. 41.5 PRLM: phone recognition followed by language modeling

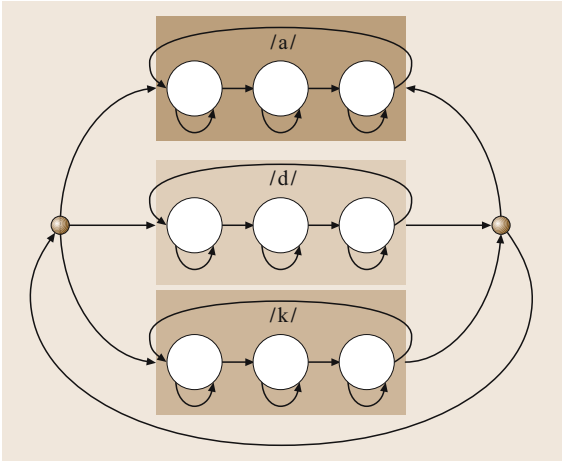


Fig. 41.7 Open phone loop (*null* grammar) with phone HMMs

tion (PLP) coefficients [41.18]). These vectors may also include dynamic features (i. e., first and second differences). Using standard assumptions (see Part E of this Handbook), the probability of this sequence with respect to the language model λ is

$$p(\boldsymbol{X}|\lambda) = \sum_{\forall \boldsymbol{s}} \prod_{t=1}^{T} p(\boldsymbol{x}_t|s_t, \lambda) \cdot P(s_t|s_{t-1}, \lambda) \,, \quad (41.15)$$

where $\boldsymbol{s} = s_1, \cdots, s_t, \cdots, s_T$ is a hypothesized sequence of states at times $t = 1, \cdots, T$. Each state s_t of the model has an observation probability $p(\boldsymbol{x}_t|s_t, \lambda)$ and a series of transition probabilities between all states and s_t. For simplicity, a bigram model $P(s_t|s_{t-1}, \lambda)$ has been selected for the state transition model. With a null grammar, the probabilities $P(s_t|s_{t-1}, \lambda)$ are uniform for fixed s_{t-1}. Typically, the observation probability $p(\boldsymbol{x}_t|s_t, \lambda)$ is modeled as a mixture of Gaussians resulting in a continuous-density HMM.

Given a sequence of observation vectors $\boldsymbol{X}$, the phone recognizer attempts to decode the maximum likelihood sequence of HMM states corresponding to $\boldsymbol{X}$:

$$\underset{\boldsymbol{s}}{\text{argmax}}\, P(\boldsymbol{s}|\boldsymbol{X}) \,. \quad (41.16)$$

Applying Bayes' rule and recognizing that the maximization is independent of $p(\boldsymbol{X})$:

$$\underset{\boldsymbol{s}}{\text{argmax}}\, \frac{p(\boldsymbol{X}|\boldsymbol{s})p(\boldsymbol{s})}{p(\boldsymbol{X})} = \underset{\boldsymbol{s}}{\text{argmax}}\, p(\boldsymbol{X}|\boldsymbol{s})P(\boldsymbol{s}) \,. \quad (41.17)$$

Equation (41.17) is the Viterbi approximation of (41.15); the goal is to maximize the following probability

$$p(\boldsymbol{X}|\boldsymbol{s}, \lambda)P(\boldsymbol{s}|\lambda) = \prod_{t=1}^{T} p(\boldsymbol{x}_t|s_t, \lambda) \cdot P(s_t|s_{t-1}, \lambda) \quad (41.18)$$

by selection of the state sequence $\boldsymbol{s}$.

The Viterbi algorithm is generally used to search the space of possible state sequences and a variety of pruning strategies can be applied to keep the search problem tractable. For language recognition, context independent phonetic units are typically used for decoding (see [41.19] as an exception). For a more-detailed description of HMM models of speech refer to Part E of this Handbook.

Phone Recognizer Training

A vital consideration in the training of phone recognizers is the availability of phonetically or orthographically labeled speech for multiple languages. These phone transcripts can be obtained from manual labeling by a trained listener or automatic labeling from a word transcript and pronunciation dictionary. As discussed later, the languages of the phone recognizers need not be those of the target languages. Phonetically transcribed corpora are relatively uncommon, and for a number of years were available only as part of the Oregon Graduate Institute multilanguage telephone speech (OGI-TS) corpus [41.20], which contained phonetically hand-labeled speech for six languages (English, German, Hindi, Japanese, Mandarin, and Spanish) collected over telephone channels. More recently, investigators have employed other corpora for training phone recognizers, including Switchboard (English) [41.21] (e.g., [41.22, 23]), CallHome (Arabic, Japanese, Mandarin, and Spanish) [41.21] (e.g., [41.22, 23]), and SpeechDat-East (Czech, Hungarian, Polish, Russian, and Slovak)d [41.24] (e.g., [41.8]).

41.3.2 Classifiers

Using the sequence of phone tokens, a number of different classification techniques can be applied to make a language recognition decision. Two popular techniques are described here in detail: PRLM, a generative model of phonetic n-gram sequences, and PR-SVM-LATT (phone recognition followed by lattice-based support vector machines), a discriminative approach to language recognition. Both approaches assume that languages differ in their phonotactic characteristics and that these differences will cause a phonetic recognition system to produce different distributions of token sequences.

PRLM

The basic PRLM decision rule is described by

$$L^* = \underset{L}{\operatorname{argmax}}\, P_L(W|\boldsymbol{X})\,, \tag{41.19}$$

where $\boldsymbol{X}$ are the acoustic observations, W is the hypothesized token sequence $W = w_1, \ldots, w_N$, and L^* is the optimal language decision. The term $P_L(W|\boldsymbol{X})$ can be decomposed using Bayes' rule:

$$P_L(W|\boldsymbol{X}) = \frac{\overbrace{p(\boldsymbol{X}|W)}^{\text{same } \forall L} \cdot P_L(W)}{\underbrace{p(\boldsymbol{X})}_{\text{same } \forall L}}\,. \tag{41.20}$$

For a given tokenizer, the terms $p(\boldsymbol{X}|W)$ and $p(\boldsymbol{X})$ are the same across language models, since $\boldsymbol{X}$ is known and W is determined from the phone recognition decoding process described above. As such, the following simplified PRLM decision rule can be applied:

$$L^* = \underset{L}{\operatorname{argmax}}\, P_L(W)\,. \tag{41.21}$$

As (41.21) suggests, only language model scores are used to make a language recognition decision. Phone recognition acts solely to discretize the input speech signal.

In real implementations $P_L(W)$ is approximated by an n-gram language model of fixed order,

$$P_L(W) \approx \prod_{i=1}^{N} \overbrace{P_L(w_i|w_{i-1}\ldots w_{i-(n-1)})}^{\text{language model}}\,. \tag{41.22}$$

Here, the probability $P_L(w_i|w_{i-1}\ldots w_{i-(n-1)}) = P_L(\hat{w}_i)$, is a look up of the frequency of occurrence of n-gram $\hat{w}_i = w_{i-n+1}w_{i-n+2}\cdots w_i$ in language L's training data. n-grams are a simple yet effective way of capturing the context preceding a token. For the language recognition problem, n-grams of order two and three (i. e., bigrams and trigrams) are commonly employed. Increasing the n-gram order should theoretically increase the ability of language recognition modeling techniques to incorporate longer term dependencies, but in practice these models are difficult to estimate given the paucity of training data. As the order n increases, the number of possible unique n-grams increases exponentially as $|\mathcal{W}|^n$, where $|\mathcal{W}|$ is the size of the tokenizer's phone inventory. Standard smoothing techniques have been shown to help mitigate the n-gram estimation problem for language recognition tasks [41.25]. Other language modeling techniques, such as binary decision trees, have also been used successfully to model n-gram sequences with $n > 3$ [41.26].

Lattice-Based PRLM. As stated above, the standard 1-best PRLM model relies solely on the probability of the phone token sequence in its decision rule. This acts as an approximation of the acoustic hypothesis space generated by the underlying HMM phonetic models for a given input.

In [41.22], a better approximation using an extension of the 1-best PRLM model is proposed in which posterior probabilities of phone tokens are incorporated into the estimation of the language models and the observation likelihoods. In this model, estimates of expected counts derived from phone lattices are used in place of 1-best counts during LM training and scoring.

To derive this extension, the standard 1-best language modeling equation can be reformulated in log form as

$$\log P_L(W) = \sum_{\forall \hat{w}} \mathrm{C}(\hat{w}) \log P_L(w_i|w_{i-1}, \ldots, w_{i-n+1})\,, \tag{41.23}$$

where

$$\hat{w} = w_i w_{i-1} \ldots w_{i-n+1} \tag{41.24}$$

and $C(\hat{w})$ is the count of the n-gram $\hat{w}$ in the sequence W. In (41.23), the sum is performed over the *unique* n-grams $\hat{w}$ in W.

In the lattice formulation, n-gram counts from the 1-best hypothesis are replaced with expected counts from a phone lattice generated by the decoding of an input message:

$$E_{\mathcal{L}}[\log P_L(W)] = \sum_{\forall \hat{w}} E_{\mathcal{L}}\left[\mathrm{C}(\hat{w})\right] \log P_L(w_i|w_{i-1}\ldots, w_{i-n+1})\,.$$

Like the 1-best hypothesis, the phone lattice is only an approximation of the acoustic hypothesis space for a given speech utterance. There is evidence that both the quality of the phonetic models and the phone lattices used for language recognition are important [41.19,27]. Many methods of generating lattices have been proposed in the automatic speech recognition (ASR) literature [41.28–30]. The Viterbi n-best technique with a fixed number of trace-backs per state is often used [41.31].

Parallel PRLM (PPRLM)

An important aspect of the PRLM method, implied by (41.21), is that the operations of tokenization (by the phone recognizer) and language modeling are decoupled. As a consequence, it is not necessary for the phone

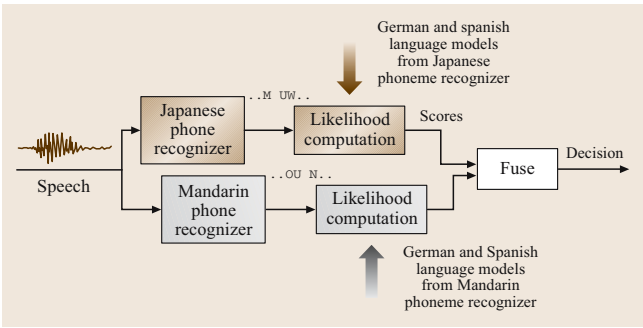


Fig. 41.8 PPRLM: *parallel* phone recognition followed by language modeling

recognizer to be trained in any of the target languages as long as the tokenization of the target languages is sufficiently distinct. In a natural extension of this observation, *Zissman* [41.15] proposed running phone recognizers for multiple languages in parallel in order to provide more-diversified coverage of the phones encountered in target languages. The resulting system configuration, commonly known as parallel PRLM (PPRLM), employs a bank of phone recognizers, each of which is followed by a set of language models trained from the corresponding token sequences that it generates. As with PRLM, the multiple phone recognizers are trained using phonetically or orthographically labeled speech, but need not be of any of the target languages. An example is shown in Fig. 41.8 where parallel Japanese and Mandarin phone recognizers are used to tokenize speech in a German versus Spanish language recognition task. The next section will describe methods by which the individual outputs are fused to produce a final score.

Language recognition systems based on PPRLM were found to outperform their competitors [41.5] and dominated the field of language recognition for a number of years. More recently, employing banks of lattice-based phone recognizers has led to further language recognition performance improvement [41.22]. Efforts have also been made to reduce the computation load of a PPRLM system by employing a single multilingual phone recognizer but to date these approaches have not proved to be more effective than PPRLM [41.32].

PPR-SVM-LATT

The PPR-SVM-LATT approach is an application of SVM classification for token streams [41.33]. For the PPR-SVM-LATT classifier, the phone token sequence for each utterance $\boldsymbol{X}$ in a language corpus is used to compute a vector of n-gram probabilities,

$$\boldsymbol{b} = \begin{pmatrix} c_1 p(\hat{w}_1|\boldsymbol{X}) \\ c_2 p(\hat{w}_2|\boldsymbol{X}) \\ \vdots \\ c_N p(\hat{w}_N|\boldsymbol{X}) \end{pmatrix} . \tag{41.25}$$

The probability $p(\hat{w}_i|\boldsymbol{X})$ represents the frequency of occurrence of n-gram $\hat{w}_i$ in the utterance $\boldsymbol{X}$, and $N = |\mathcal{W}|$ is the number of possible n-grams. The n-gram probabilities can be computed from counts from the 1-best phone sequence or from expected counts from phone lattices. Experiments have determined that expected counts from lattices produce better performance.

The values c_i in (41.25) are weights that normalize the components of the vector to a common scale. Typically, a weighting such as the term-frequency log likelihood ratio (TFLLR) [41.33] can be applied. For TFLLR,

$$c_i = \frac{1}{\sqrt{p(\hat{w}_i|\{\boldsymbol{X}_j\})}} , \tag{41.26}$$

the probability $p(\hat{w}_i|\{\boldsymbol{X}_j\})$ is calculated across all utterances of all languages in the training corpus.

After converting utterances to vectors, a simple linear kernel, $K(\boldsymbol{b}_1, \boldsymbol{b}_2) = \boldsymbol{b}_1^{\mathrm{T}} \boldsymbol{b}_2$, is used, where the $\boldsymbol{b}_i$ are the expansion of two utterances as in (41.25). Training and scoring are done in a manner analogous to the cepstral SVM.

41.4 System Fusion

The previous sections of this chapter described a variety of methods available for implementing a language recognition system. In practice, additional performance improvements can be obtained by running multiple systems in parallel and fusing the individual scores using a back-end. If the recognizers do not behave uniformly (i. e., their errors are not completely correlated) then combining the scores may make it possible to exploit complementary information that exists in the scores. Score fusion was first adopted as a means of combining the scores produced by a PPRLM language recognizer and has since been applied to fusing scores of differ-

ent types of recognizers, such as those described in Sects. 41.2 and 41.3.

41.4.1 Methods

For purpose of this discussion it is convenient to divide approaches to fusion into two types, rule based and classifier based. Rule-based fusion applies a fixed formula to the scores, such as a weighted-sum rule or product rule, where weights are either uniform or are chosen empirically using a set of development data. One example of a rule-based method for combining the scores in a PPRLM system will be discussed below. The second approach to fusion utilizes a development data set to train a classifier from a feature vector constructed from the individual system scores. In theory, the joint statistics among the scores (elements of the feature vector) can be exploited by the classifier to produce overall performance that is superior to that achieved with a rule-based back.end.

Product-Rule Fusion

A method for fusing the scores of a PPRLM language recognition system using the product rule was proposed by *Zissman* [41.5]. Assume a PPRLM language recognizer that uses K phone recognizers to detect L languages. For each test input, a set of KL (linear) scores is generated, with each phone recognizer producing L scores. For an input utterance $\boldsymbol{X}$, single per-language scores $y(\boldsymbol{X}|l)$ are computed by first normalizing the likelihoods $s_{k,l}$ produced by phone recognizer k and then multiplying the normalized per-language values across all phone recognizers:

$$y(\boldsymbol{X}|l) = \prod_{k}^{K} \frac{s_{k,l}}{\sum_{l}^{L} s_{k,l}} \,. \tag{41.27}$$

The normalization step puts scores across phone recognizers into a common range, and the multiplication step creates a final score. Probabilistically, the first step turns the raw phone recognizer scores into posteriors (with an assumption of equal priors) and the second step computes a joint probability of the observed data under the assumption that the information sources (individual PRLM systems) are independent. A potential drawback of product-rule fusion is that a single score close to zero may have an undesirably large impact on the final result. Despite the apparent inaccuracy of the independence assumption, the product-rule method proved to be an effective way of combining PPRLM scores [41.5].

Classifier Fusion

Classifier-based fusion for PPRLM language recognition systems was originally proposed by *Yan* and *Bernard* [41.34] using neural networks and by *Zissman* [41.35] using Gaussian classifiers, and these authors reported that performance was superior to that obtained with the product-rule approach. A block diagram of a Gaussian-based fusion approach for language recognition is shown in Fig. 41.9. The method consists of linear discriminant analysis (LDA) normalization followed by Gaussian classification. Although there can be no guarantee that this particular classifier-based fusion architecture produces optimum results in all language recognition applications, extensive development testing has shown that the technique produces reliably superior performance. Other investigators have reported results using neural network-based back-end fusion [41.22].

The generic block diagram in Fig. 41.9 shows a set of core language recognizers (e.g., GMM, SVM, PRLM, PPRLM), each of which produces a score or score vector for each language hypothesis given an input utterance. These scores are used to form the elements of a feature vector that is transformed via linear discriminant analysis, a method by which the feature vector is projected into a reduced dimensional space in such a way as to maximize interclass separability and minimize intraclass variability. In the figure, the transformed space is of dimension $L-1$, where L is the number of classes (languages) represented in the back-end training data. The transformed vector also has the desirable property that its features are decorrelated. It is important to recognize that the individual recognizers may produce different sized score vectors and that the scores may represent likelihoods of any languages, not just the target languages.

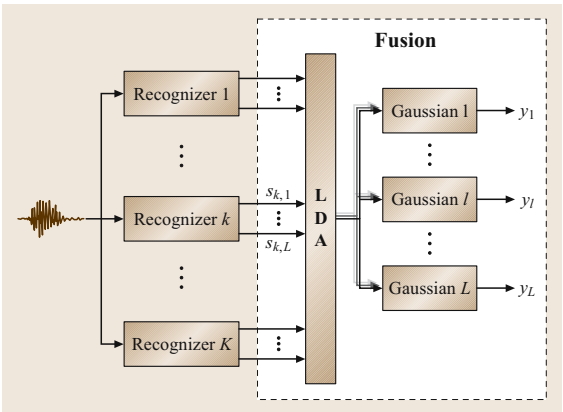


Fig. 41.9 Gaussian-based fusion for language recognition

The transformed feature vectors are then processed by a parallel bank of Gaussian classifiers. Each Gaussian has input dimension $L-1$, a diagonal grand covariance (i. e., pooled across all the back-end training data), and produces an output $y(\boldsymbol{X}|l)$.

41.4.2 Output Scores

The form of the final system scores depends on the nature of the language recognition application. For closed-set identification, the index of the maximum score, $\max_l y(\boldsymbol{X}|l)$, produced using either the rule- or classifier-based method is sufficient to identify the winning language hypothesis. For a detection task, where decisions are made using a fixed threshold, the output score must be formed in such a way as to be consistently meaningful across utterances. Typically, this is accomplished by interpreting the (linear) scores $y(\boldsymbol{X}|l)$ as likelihoods and forming an estimated likelihood ratio $r(\boldsymbol{X}|l)$ between the target language score and the sum (or average) of the others:

$$r(\boldsymbol{X}|l) = \frac{y(\boldsymbol{X}|l)}{\frac{1}{L-1}\sum_{j\neq l}^{L} y(\boldsymbol{X}|j)} \,. \tag{41.28}$$

The estimated likelihood ratio can then be used to set a threshold to achieve a specific application dependent operating point.

Language recognition applications often require that scores be reported as posterior probabilities rather than as likelihoods or likelihood ratios. This may be the case when the downstream consumer (human or machine) requires confidence scores or scores that have been mapped from unconstrained likelihoods or likelihood ratios to a normalized range (0–1 or 0–100). In principle, the scores $y(\boldsymbol{X}|l)$ produced by the back-end fusion for input utterance $\boldsymbol{X}$ can be used directly to generate posterior probabilities $P(l)$ for each target language l via Bayes' rule:

$$P(l|\boldsymbol{X}) = \frac{y(\boldsymbol{X}|l)P_l}{\sum_l^L y(\boldsymbol{X}|l)P_l} \,, \tag{41.29}$$

where P_l represents the prior probability of target language l. In practice, posteriors estimated this way may be unreliable due to the inaccurate assumptions made in constructing the overall system. Rather than treating the scores as true likelihoods, it is preferable to view them as raw scores and to estimate posteriors using another classifier.

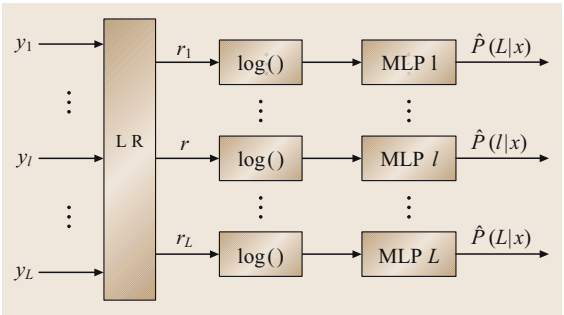


Fig. 41.10 Posterior probability estimation

A block diagram of a method that has been found useful for posterior probability estimation is shown in Fig. 41.10. The fusion back-end log-likelihood ratio estimates $\log r(\boldsymbol{X}|l)$ are treated as raw inputs to a bank of language-dependent single-input logistic regression classifiers (implemented using LNKnet [41.36] as multilayer perceptrons) that produce estimates $\hat{P}(l|\boldsymbol{X})$ of the posterior probabilities. These estimates can then be interpreted by humans as meaningful measures of confidence.

41.5 Performance Assessment

The language recognition approaches described in this chapter have been subject to extensive development, and this section describes their evaluation under conditions that follow protocols established by the US National Institute of Standards and Technology (NIST). Since 1996, NIST has coordinated several evaluations with the goal of assessing the existing state-of-the-art in language recognition technology. The most recent language recognition evaluation (LRE 2005) protocol will form the basis of the performance results presented in this section. A description of the 2005 NIST language recognition evaluation plan can be found in [41.37].

41.5.1 Task and Corpus

The task for LRE 2005 was the detection of the presence of one of seven target languages (English, Hindi, Japanese, Korean, Mandarin, Spanish, Tamil) in telephone speech utterances with nominal durations of 30, 10, and 3 seconds selected from unspecified lan-

guages and sources. The evaluation material contained 3,662 segments for each nominal duration. Although NIST evaluated and reported performance results under several conditions, its primary evaluation condition consisted of trials for which the segments had 30 second durations, were from one of the seven target languages, and were from material collected by the Oregon Health Sciences University (OHSU). Unless otherwise stated, the results presented here will be for the NIST primary evaluation condition. For a complete description of LRE 2005, see [41.38].

Training data for the systems described below was drawn from four telephone speech corpora: CallFriend, CallHome, Mixer, and Fisher, and cover 13 languages (for more information on these corpora, see [41.21]). CallFriend and CallHome contain conversations between acquaintances, while Mixer and Fisher contain conversations between strangers. For a detailed description of the use of this material for training the recognizers and the fusion back-end, see [41.23].

41.5.2 Systems

The fully fused system contained five core language recognizer components. For each recognizer, nonspeech frames were removed from the train and test utterances using a GMM-based speech activity detector.

GMM

The Gaussian mixture model (GMM)-based language recognizer used 2048 mixture components, shifted delta cepstral (SDC) coefficients, feature normalization across utterances to zero mean and unit variance on a per-feature basis, and feature mapping [41.3]. The SDC parameters N-d-P-k were set to 7-1-3-7 (Sect. 41.2.1), resulting in a 49-dimension feature vector computed over a 21-frame (210 ms) time span. In addition, gender-dependent language models were trained for each of the 13 languages in the training material. A background model was trained by pooling all the training material, and gender-dependent target language models were trained by adapting from the background model. This method of training permits fast scoring during recognition and was proposed by [41.39] for language recognition. The GMM system produced a total of 26 scores per test utterance.

SVM

The support vector machine (SVM)-based language recognizer used a generalized linear discriminant sequence (GLDS) kernel, expansion into feature space using monomials up to degree three, and SDC coefficients derived as for the GMM recognizer. The features were normalized across utterances to zero mean and unit variance on a per-feature basis. Thirteen language models were constructed from the training material and 13 scores were produced per test utterance.

PPRLM

The PPRLM-based language recognizer used six OGI-trained phone recognizers, a 39-dimensional MFCC feature vector containing cepstra, delta, and double delta coefficients, and trigram language modeling. Each phone recognizer's token sequences were used to train 13 language models, resulting in an output feature vector of dimension 78. The phone recognizers and language models were trained using Cambridge University engineering department's hidden Markov model toolkit (HTK) [41.40].

PPRLM-LATT

A PPRLM based language recognizer using lattices to generate probabilistic counts of phone frequencies for bigram and trigram language modeling. Phone recognizers were trained using material from CallHome (Arabic, Japanese, Mandarin) and Switchboard English (landline and cellular). A total of seven phone recognizers were trained, resulting in 91 (7×13) scores per input utterance. The tokenizers were selected based on results of development testing conducted in conjunction with LRE 2005 ([41.41]).

PPR-SVM-LATT

The PPR-SVM-LATT-based language recognizer used lattices to generate probabilistic counts of phone frequencies that are then classified with support vector machines. Probability vectors were constructed from unigrams and bigrams, and a linear kernel with TFLLR weighting was applied. Lattices of phones were produced using HMM models trained on six OGI languages, as in PPRLM. A total of 78 (6×13) scores were produced per input utterance.

The scores of the core language recognizers were stacked into a 286-dimension feature vector and fed to the fusion back-end for dimension reduction (linear discriminant analysis) and classification (seven Gaussians with pooled diagonal covariances). The outputs of the back-end were treated as estimated target language likelihoods and log likelihood ratios were formed by taking the log of the quantity in (41.28). These scores were treated as the final outputs of the system.

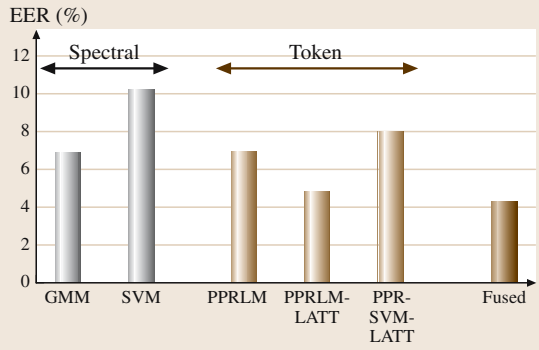


Fig. 41.11 Performance (percentage EER) of spectral, token, and fused language recognition systems on the 2005 NIST LRE primary condition test

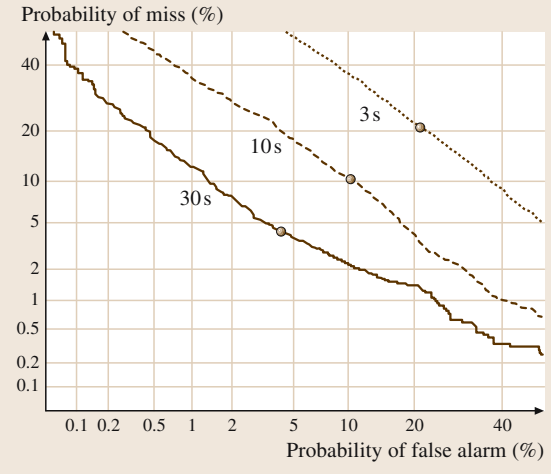


Fig. 41.12 Performance of five-system fusion on the LRE 2005 OHSU target language test segments for durations 30 s (*solid line*), 10 s (*dashed line*), and 3 s (*dotted line*). Filled circles indicate equal error points

41.5.3 Results

NIST language recognition evaluations require participants to provide two outputs for each test utterance, a hard decision and an arbitrarily scaled likelihood score. The hard decisions are used by NIST to evaluate a cost function while the scores are used to sweep out a detection error trade-off (DET) curve [41.42], a variant of the familiar receiver operating characteristic (ROC) curve. To compare detection performance across competing systems, it is convenient to use a summary statistic such as the equal error rate (EER), and this statistic has been adopted for comparison of the language recognition systems described in this section. Results for the NIST primary condition described above, given in percentage EER, are shown in Fig. 41.11. The recognizers have been categorized according to whether they are spectral (GMM or SVM) or token based (PPRLM, PPRLM-LATT, PPR-SVM-LATT). Individually, the best performing recognizer is PPRLM-LATT, while the SVM system produces the most errors. The remaining systems (GMM, PPRLM, and PPR-SVM-LATT) perform comparably. It is worth noting that fusing the scores of the spectral systems (SVM and GMM) results in a large performance gain [41.23]. Fusing the scores of all five recognizers produces the best overall result.

Figure 41.12 shows the DET plots for five-recognizer-system fusion for the LRE 2005 OHSU target language test segments with durations 30, 10, and 3 seconds. Table Table 41.1 gives the corresponding EERs for these DET curves. It is apparent that language recognition performance becomes increasingly challenging as the test segment duration decreases, although recent work has demonstrated that performance at shorter durations can be substantially improved [41.8].

41.5.4 Computational Considerations

Although the focus of the discussion of language recognition systems has been performance, any practical implementation of such a system needs to account for computational complexity. A comparison of process-

Table 41.1 Performance, measured as percentage equal error rate (EER) of full-fusion system for 30 s, 10 s, and 3 s test utterances

Duration	EER(%)
30	4.3
10	10.3
3	21.2

Table 41.2 Processing speed of selected language recognizers as measured with a 3 GHz Xeon processor with 2 GB RAM running Linux. The recognition task involved 12 target languages and five minute audio file. The PPRLM system contained six OGI-trained phone recognizers (no lattices). All factors refer to speeds faster than real time (RT)

Reognizer	Speed
GMM	17 RT
SVM	52 RT
PPRLM	2 RT

ing speed for three types of recognizers is shown in Table 41.2. Note that all factors indicate the degree to which the algorithms are *faster* than real time. All measurements were made using 12 language models and a single 5 minute audio file. The platform was a 3 GHz Xeon processor with 2 GB RAM and Linux OS. The GMM, SVM, and PPRLM systems were described in previous sections of this chapter. The PPRLM language recognizer used six OGI tokenizers and no lattices.

41.6 Summary

This chapter has presented a variety of methods for implementing automatic language recognition systems. Two fundamental approaches, based on either the spectral or phonotactic properties of speech signals, were successfully exploited for the language recognition task. For the spectral systems, these included GMM and SVM modeling using frame-based shifted delta cepstral coefficients. For phonotactic systems, variations of the classic parallel phone recognition followed by language modeling were described. Performance gains can be achieved by fusing the scores of multiple systems operating simultaneously, and a framework for implementing classifier based fusion was described. Finally, performance results were presented for the spectral, token, and fused systems using the standardized protocols developed by NIST. For the latest advances in the field of automatic language recognition, the reader is referred to the most recent proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), the Odyssey Speaker and Language Recognition Workshop, and the International Conference on Spoken Language Processing (Interspeech – ICSLP).

References

41.1 S. Davis, P. Mermelstein: Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences, IEEE Trans. Acoust. Speech Signal Process. (1980) pp. 357–366

41.2 H. Hermansky, N. Morgan, A. Bayya, P. Kohn: Compensation for the Effect of the Communication Channel in Auditory-Like Analysis of Speech, Proc. Eurospeech (1991) pp. 1367–1371

41.3 D.A. Reynolds: Channel robust speaker verification via feature mapping, Proceedings of the, International Conference on Acoustics Speech and Signal Processing (2003) pp. II–53, –56

41.4 P.A. Torres-Carrasquillo, E. Singer, M.A. Kohler, R.J. Greene, D.A. Reynolds, J.R. Deller Jr.: Approaches to language identification using Gaussian mixture models and shifted delta cepstral features, International Conference on Spoken Language Processing (2002) pp. 89–92

41.5 M. Zissman: Comparison of Four Approaches to Automatic Language Identification of Telephone Speech, IEEE Trans. Speech Audio Process. **4**(1), 31–44 (1996)

41.6 A. Dempster, N. Laird, D. Rubin: Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. **39**, 1–38 (1977)

41.7 D.A. Reynolds, T.F. Quatieri, R. Dunn: Speaker Verification Using Adapted Gaussian Mixture Models, Digital Signal Process. **10**(1-3), 19–41 (2000)

41.8 P. Matějka, L. Burget, P. Schwarz, J. Černocký: Brno University of Technology System for NIST 2005 Language Recognition Evaluation, Proc. IEEE 2006 Odyssey: The Speaker and Language Recognition Workshop (2006)

41.9 V.N. Vapnik: *Statistical Learning Theory* (Wiley, New York 1998)

41.10 R. Collobert, S. Bengio: SVMTorch: Support Vector Machines for Large-Scale Regression Problems, J. Mach. Learn. Res. **1**, 143–160 (2001)

41.11 W.M. Campbell: Generalized Linear Discriminant Sequence Kernels for Speaker Recognition, Proceedings of the International Conference on Acoustics Speech and Signal Processing (2002) pp. 161–164

41.12 J.C. Platt: Probabilities for SV Machines. In: *Advances in Large Margin Classifiers*, ed. by A.J. Smola, P.L. Bartlett, B. Schölkopf, D. Schuurmans (MIT, Cambridge 2000) pp. 61–74

41.13 W.M. Campbell, J.P. Campbell, D.A. Reynolds, E. Singer, P.A. Torres-Carasquillo: Support vector machines for speaker and language recognition, Comput. Speech. Lang. **20**(2-3), 210–229 (2006)

41.14 P.A. Torres-Carrasquillo, D.A. Reynolds, J.R. Deller Jr.: Language Identification Using Gaussian Mixture Model Tokenization, Proceedings of the International Conference on Acoustics Speech and Signal Processing (2002) pp. 757–760

41.15 M. Zissman, E. Singer: Automatic Language Identification of Telephone Speech Messages Using Phoneme Recognition and n -Gram Modeling, Proceedings of the International Conference on

Acoustics Speech and Signal Processing (1994) pp. 305–308
41.16 Y. Yan, E. Barnard: An Approach to Automatic Language Identification Based on Language-Dependent Phone Recognition, Proceedings of the International Conference on Acoustics Speech and Signal Processing (1995) pp. 3511–3514
41.17 A. House, E. Neuberg: Toward automatic identification of the language of an utterance. I, Preliminary methodological considerations, J. Acoust. Soc. Am. **62**, 708–713 (1977)
41.18 H. Hermansky: Perceptual linear predictive (PLP) analysis for speech, J. Acoust Soc. Am. **87**, 1738–1752 (1990)
41.19 D. Zhu, M. Adda-Decker, F. Antoine: Different Size Multilingual Phone Inventories and Context-Dependent Acoustic Models for Language Identification, Proc. Interspeech (2005) pp. 2833–2836
41.20 Y.K. Muthusamy, R.A. Cole, B.T. Oshika: The OGI Multi-language Telephone Speech Corpus, International Conference on Spoken Language Processing (1992) pp. 895–898
41.21 Linguistic Data Consortium, http://www.ldc.upenn.edu/ (2007)
41.22 J.L. Gauvain, A. Messaoudi, H. Schwenk: Language Recognition Using Phoneme Lattices, International Conference on Spoken Language Processing (2004) pp. 2833–2836
41.23 W. Campbell, T. Gleason, J. Navritil, D. Reynolds, W. Shen, E. Singer, P. Torres-Carrasquillo: Advanced Language Recognition using Cepstra and Phonotactics: MITLL System Performance on the NIST 2005 Language Recognition Evaluation, Proc. IEEE 2006 Odyssey: The Speaker and Language Recognition Workshop (2006)
41.24 Eastern European Speech Databases for Creation of Voice Driven Teleservices, http://www.fee.vutbr.cz/SPEECHDAT-E/ (2007)
41.25 E. Singer, P.A. Torres-Carrasquillo, T.P. Gleason, W.M. Campbell, D.A. Reynolds: Acoustic, Phonetic, and Discriminative Approaches to Automatic Language Identification, Proceedings of Eurospeech (2003) pp. 1345–1348
41.26 J. Navratil: Recent advances in phonotactic language recognition using binary-decision trees, International Conference on Spoken Language Processing, Vol. 2 (2006)
41.27 P. Matějka, P. Schwarz, J. Černocký, P. Chytil: Phonotactic Language Identification using High Quality Phoneme Recognition, Proceedings of Interspeech (2005) pp. 2833–2836
41.28 F. Weng, A. Stolcke, A. Sankar: New Developments in Lattice-based Search Strategies in SRI's H4 system, Proceedings of DARPA Speech Recognition Workshop (1998) p. 100
41.29 H. Ney, X. Aubert: A word graph algorithm for large vocabulary, continuous speech recognition, International Conference on Spoken Language Processing (1994) pp. 1355–1358
41.30 F. Weng, A. Stolcke, A. Sankar: Efficient Lattice Representation and Generation, International Conference on Spoken Language Processing (1998) pp. 100–100
41.31 R. Schwartz, S. Austin: A Comparison of several approximate algorithms for finding multiple (n-best) sentence hypotheses, Proceedings of the International Conference on Acoustics Speech and Signal Processing (1991) pp. 701–704
41.32 J. Navratil: Automatic Language Identification. In: *Multilingual Speech Processing*, ed. by T.S. Pittsburgh, K. Kirchhoff (Academic, New York 2006)
41.33 W.M. Campbell, J.P. Campbell, D.A. Reynolds, D.A. Jones, T.R. Leek: High-Level Speaker Verification with Support Vector Machines, Proceedings of the International Conference on Acoustics Speech and Signal Processing (2004) pp. I–73–I–76
41.34 Y. Yan, E. Barnard: Experiments for an Approach to Language Identification with Conversational Telephone Speech, Proceedings of the International Conference on Acoustics Speech and Signal Processing (1996) pp. 789–792
41.35 M.A. Zissman: Predicting, Diagnosing and Improving Automatic Language Identification Performance, Proceedings of Eurospeech (1997) pp. 51–54
41.36 R. Lippman, L. Kukolich: LNKnet Pattern Recognition Software Package, http://www.ll.mit.edu/IST/lnknet/ (2007)
41.37 A. Martin, G. Doddington: The 2005 NIST Language Recognition Evaluation Plan, http://www.nist.gov/speech/tests/lang/2005/LRE05EvalPlan-v5-2.pdf (2005)
41.38 A.F. Martin, A.N. Le: The Current State of Language Recognition: NIST 2005 Evaluation Results, Proc. IEEE 2006 Odyssey: The Speaker and Language Recognition Workshop (2006)
41.39 E. Wong, S. Sridharan: Methods to improve Gaussian mixture model based language identification system, International Conference on Spoken Language Processing (2002) pp. 93–96
41.40 Hidden Markov Model Toolkit, http://htk.eng.cam.ac.uk/ (2007)
41.41 W. Shen, W. Campbell, T. Gleason, D. Reynolds, E. Singer: Experiments with Lattice-based PPRLM Language Identification, Proc. IEEE 2006 Odyssey: The Speaker and Language Recognition Workshop (2006)
41.42 A. Martin, G. Doddington, T. Kamm, M. Ordowski, M. Przybocki: The DET Curve in Assessment of Detection Task Performance, Proceedings of Eurospeech (1997) pp. 1895–1898

39. Principles of Spoken Language Recognition

C.-H. Lee

In this introductory chapter to Part G of this Handbook on spoken language recognition, we provide a brief overview of the principles of state-of-the-art language recognition approaches, and a general discriminative training framework to improve the performance and robustness of language recognition systems. It is followed by three chapters. The first of these addresses issues related to spoken language characterization in which knowledge sources can be utilized to distinguish one language from another. The second chapter deals with language identification based on phone recognition followed by language modeling using either spectral or token-based approaches. The third chapter presents vector-space characterization approaches to converting speech utterances into spoken document vectors for modeling and classification. With recent progress in speech processing, machine learning, and text categorization, we expect significant technology advances in spoken language recognition in the years to come.

This chapter is organized as follows. Section 39.2 briefly describes the principle of spoken language recognition. Sections 39.3 and 39.4 formulate the popular parallel phone recognition followed by language modeling (P-PRLM) and vector-space characterization (VSC) approaches to spoken language identification. In Sect. 39.5 we extend these formulations to spoken language verification. Finally a general discriminative training framework for non-support vector machine (non-SVM) classifiers is presented in Sect. 39.6, followed by a brief summary in Sect. 39.7.

39.1 Spoken Language

A spoken language can be identified using information from a number of sources. When human beings are constantly exposed to a language without being given any linguistic knowledge, they learn to discriminate subtle differences by perceiving speech cues in the specific languages. Nonetheless lexical knowledge is usually the main source of information for human listeners, especially when the languages to be distinguished are similar. For automatic spoken language recognition by machines, state-of-the-art systems often consider unknown utterances as a concatenation of signal patterns, and probabilistic modeling and classification techniques are then adopted to characterize these patterns and their corresponding language identities. Although lexical models are not explicitly utilized, models of phones in a subset of languages are often used to decode speech for further processing. Most human listeners are usually constrained by their surroundings from effectively learning a large number of languages. On the other hand machines are equipped to take in virtually an unlimited amount of speech and text. It is interesting to note that advanced machines therefore have the potential to surpass human performance in spoken language recognition.

With a coordinated effort between government funding agencies and technical communities for data collection and benchmark performance evaluation, the current availability of large collections of speech examples from a selected set of languages, easy access to fast and affordable computing equipment, and recent

advances in speech modeling and machine learning, many new algorithms have recently been explored in this area. We are now witnessing rapid progress in the area of spoken language modeling and recognition technologies.

In this introductory chapter to Part G of this Handbook on spoken language recognition, we provide a brief discussion of the principles of state-of-the-art language recognition approaches. It is then followed by three more chapters. The first addresses issues of spoken language characterization, in which knowledge sources that can be utilized to distinguish one language from another are presented. Contrasts between human and machine language recognition are made to highlight that language cues perceived by human listeners can be incorporated into current technologies to improve machine capabilities. It is also noted that there are a few orders of magnitude more languages and dialects in existence than current systems are equipped to handle. Many of these languages are rare and extensive data collection to construct recognizers for these languages can be prohibitive. Some of today's prevailing technologies may have to be modified to alleviate these limitations. New paradigms may also have to be established to examine the language recognition problem from a completely different perspective.

The two other chapters in this part of the Handbook describe prevailing trends in current language recognition system design. The first attempts to divide language recognition approaches into two broad categories. The first method is spectral-based methods, in which each spoken utterance or segment is represented by a sequence of feature vectors that are often short-time spectral representation of speech frames. This is similar to what is typically done in the front-end feature-extraction module of an automatic speech recognition (ASR) [39.1] system. These spectral vectors are assumed to be generated from different source languages, and therefore have different characteristics. A collection of models, one for each language to be considered, can then be built with labeled spectral vector examples collected from all the languages. These models can be probabilistic, such as a Gaussian mixture model (GMM) [39.2], or vector based, such as a support vector machine (SVM) [39.3]. It can be seen that this spectral-based framework is purely acoustic, while no linguistic information, such as phones or words, is used. The second method is called the token-based approach, in which an intermediate set of speech units, also referred to as tokens, is used to represent speech. An utterance is first decoded and segmented into a sequence of such tokens. These token streams are then used to extract features, designing classifiers, and performing spoken language classification.

The last chapter in this part of the Handbook represents a new vector-space characterization (VSC) [39.4] approach to language recognition in which an utterance is considered as a spoken document. A term-document matrix representation of all utterances in a training set can then be established for indexing and retrieval purposes. The terms here can refer to acoustic words (AWs), which are sequences of acoustic letters (alphabets), or fundamental sound units, and the vector elements are often co-occurrence statistics describing the frequencies of AWs in each spoken utterance. Language recognition with an unknown spoken query can then be treated as a document-retrieval problem similar to that which is commonly faced in the information retrieval (IR) [39.5] community. It can also be cast as a text categorization (TC) [39.6, 7] problem in which all training documents are used to build text categorizers, or topic classifiers, one for each language to be considered. In so doing language recognition can be accomplished by comparing the unknown spoken query vector with each of the language classifiers to make a recognition decision. This vector-based modeling framework allows one to represent speech utterances with a very high-dimension document vector, sometime in the tens of thousands, so that many of the feature extraction and classifier learning algorithms currently available in the IR and TC literatures can be easily adopted for spoken language recognition. New language cues can also be exploited and incorporated into the spoken document vectors, along the same lines as a recently proposed automatic speech attribute transcription (ASAT) [39.8] paradigm for ASR, to improve system performance.

39.2 Language Recognition Principles

Automatic spoken language identification (LID) is a process of determining the language spoken in a speech sample. LID technology is needed in many applications such as multilingual conversational systems [39.9], spoken language translation [39.10], multilingual speech recognition [39.11], and spoken document re-

trieval [39.12]. It is also a topic of great importance in the areas of intelligence and security, where the language identities of recorded messages and archived materials need to be established before any information can be extracted from them. In the past few decades, researchers have explored many speech and language knowledge sources, including articulatory parameters [39.13], acoustic features [39.14], prosody [39.15,16], phonotactic [39.17,18], and lexical knowledge [39.19]. Recently, investigators have reported promising results using *shifted-delta-cepstral* acoustic features [39.20]. From the perspective of human language recognition, humans constantly exposed to a language without being given any linguistic knowledge learn to determine the identity of the language by detecting some distinct speech cues in the specific language. For example, an English-speaking listener can often appreciate the tonal nature of Mandarin Chinese. It is also noted in human perceptual experiments that listeners with a multilingual background often perform better than monolingual listeners in identifying unfamiliar languages [39.21]. These reasons motivate us to explore useful speech attributes in languages, along the same lines as a recently proposed ASAT paradigm for automatic speech recognition (ASR) [39.8]. Other useful speech and language features, such as prosodic and syllabic content, can also be incorporated into this feature vector.

In a typical pattern recognition setting, one must evaluate the a posterior probability, $P(\lambda_l|X)$, of the l-th language to be considered with a model λ_l given an unknown utterance X. To make a decision, the language that yields the maximum $P(\lambda_l|X)$ is usually identified as the target language. Many algorithms developed in automatic speech and speaker recognition [39.22, 23] can be adopted and extended to language recognition. Recent advances in acoustic and language modeling [39.24] have also contributed to progress in language recognition.

Taking advantage of recent advances in continuous speech recognition with hidden Markov models HMMs [39.25], probabilistic approaches [39.26–29] have been developed by exploiting techniques in acoustic phone modeling and n-gram language modeling. These characterize a spoken language using probability distributions of spectral features in the form of linguistically defined symbols, such as phones and syllable-like units [39.17, 30], where phone models are used to decode speech utterances into sequences of such fundamental symbols. An interpolated phone-based n-gram language model is then constructed for each language, and to derive phonotactic scores. Such an example is parallel phone recognition followed by language modeling (P-PRLM) [39.18], which uses multiple single-language phone recognizers as a front end and language-dependent language models as the back end. The phonotactic approach has been shown to provide superior performance on NIST language recognition evaluation (LRE) tasks [39.27]. It is generally agreed that the fusion of multiple phonotactic features improves performance. For example, the P-PRLM approach derives multiple sets of phonotactic features. Others have found that multiresolution phonotactic analysis, such as phone unigram, bigram, and trigram approaches, complement each other [39.17, 31, 32].

A key question here is whether a phone definition is needed to represent fundamental speech units. If we can *tokenize* speech with a manageable set of *spoken letters* and develop models to decode spoken utterances into sequences of these acoustic units, it is clear that the statistics of these spoken letters and their co-occurrences can be used to discriminate one spoken language from another. Although common sounds are shared considerably across spoken languages, the statistics of these sounds, such as phone n-grams, can differ considerably from one language to another. An interesting generalization through acoustic units is to represent any spoken utterance (also referred to as a spoken document when presented in the form of *spoken letters*) with a high-dimensional feature vector, where each element carries sound co-occurrence statistics. This is similar to a latent semantic indexing (LSI) vector representation [39.33] of text documents, which is commonly used in information retrieval (IR) systems [39.5]. Such statistics are considered to be salient features for indexing and retrieving documents. We call this paradigm vector-space characterization of speech [39.4].

To develop a universal set of fundamental speech units to cover the acoustic characterization of all spoken languages we relax the notion of language-specific phonetic definitions. To relate this universal unit set to language discrimination, it is well known that the entropy of English can be effectively reduced when high-order statistics of letters are computed [39.34]. For example, of the 26 English letters plus the space character, a few of them, such as n, s, and t, occur more often in text than others, such as x and q. By incorporating this first-order statistics, or unigram, the entropy of English text can easily be reduced from 4.76 to 4.03 bits. When letter bigrams and trigrams are added, the entropy is further reduced. This set of statistics can be used to discriminate among languages already decoded, even

if no extra dictionary information is explicitly utilized. The same notion can be extended to spoken language identification. This is sometimes referred to as a bag-of-sound representation [39.26], in analogous to the bag-of-words representation of text documents used in IR. Just as in P-PRLM multiple bags of sound can also be used [39.35].

A model corresponding to the acoustic letters approach mentioned above is called an acoustic segment model (ASMs) [39.36] and can be used to decode spoken utterances into strings of such units. HMMs [39.25] are often used to model the collection of ASMs, which can be established in a bottom-up unsupervised manner. ASMs have been used to construct an acoustic lexicon for isolated word recognition with high accuracy [39.36]. Acoustic words (AWs) can now be formed by grouping adjacent acoustic letters, and serve as a basis to build feature vectors for spoken documents and build language classifiers. Therefore, automatic language classification can be formulated as a text categorization (TC) [39.6] problem based on LSI-derived feature vectors and discriminative classifier design [39.37]. Many existing feature representation and machine learning techniques widely available in the IR and TC literature can now be adopted.

39.3 Phone Recognition Followed by Language Modeling (PRLM)

Consider a speech utterance X to be represented by a sequence vectors of length τ, $O = \{\boldsymbol{o}_1, \ldots, \boldsymbol{o}_i, \ldots, \boldsymbol{o}_\tau\}$, in which $\boldsymbol{o}_i$ is a feature vector extracted from X at time i. We can express the a posteriori probability of language l using Bayes theorem, as follows:

$$P(l|O) = P(O|l)\frac{P(l)}{P(O)}\,, \tag{39.1}$$

where $P(l)$ and $P(O)$ are prior probabilities of observing language l and vector sequence O, respectively. Without loss of generality we can assume $P(l)$ to be equal for all languages. The language-independent term, $P(O)$, usually does not affect our decision rules. They will be dropped hereafter in this chapter. Now we can apply the *maximum a posteriori* (MAP) decision rule for LID as:

$$\hat{l} = \arg\max_l P(l|O) = \arg\max_l P(O|\lambda_l)\,, \tag{39.2}$$

where λ_l is an model for the l-th language. Here the model λ_l can be any reasonable characterization of the feature vectors. A straightforward choice is to consider all such vectors to be generated from a language-specific density, and a GMM can then be used to model the source. This is exactly the same formulation as in GMM-based text-independent speaker identification [39.38]. This approach is purely frame based, and no segmental information is used. SVMs [39.39] have been used to design both high-performance GMM-based speaker and language recognition systems [39.2, 3, 40].

If we fold in some fundamental speech units, and associate a given utterance with a sequence of such units then more-detailed models can be incorporated. For example, if we have a set of phone HMMs λ_l^{AM} and a set of phone language models λ_l^{LM} trained with speech data with implied phone sequence labels from language l, then we have

$$\hat{l} = \arg\max_l \sum_{\forall q} P\left(O|q, \lambda_l^{\mathrm{AM}}\right) P\left(q|\lambda_l^{\mathrm{LM}}\right)\,, \tag{39.3}$$

where q is a candidate phone sequence. In many cases the sum in (39.3) is approximated by finding the most dominant phone sequence $\hat{q}_l$ with the l-th phone model λ_l^{AM} using Viterbi decoding,

$$\hat{q}_l = \arg\max_{\forall q} P\left(O|q, \lambda_l^{\mathrm{AM}}\right)\,, \tag{39.4}$$

and solving for the following:

$$\hat{l} \approx \arg\max_l \left[P\left(O|\hat{q}_l, \lambda_l^{\mathrm{AM}}\right) P\left(\hat{q}_l|\lambda_l^{\mathrm{LM}}\right)\right]. \tag{39.5}$$

This is known as the phone recognition followed by language modeling (PRLM) approach to LID. Since obtaining high-accuracy phone and language models for each individual language under consideration is not always possible, one can consider training a single set of phone models to cover all languages and use it to decode all spoken utterances. Although the decoding performance for sounds not properly modeled by this model set is usually not good, we can still get reasonable performance by dropping the dependency on l for the acoustic score term and the phone sequence. Now (39.5) is solved by:

$$\hat{l} \approx \arg\max_l P\left(\hat{q}|\lambda_l^{\mathrm{LM}}\right)\,. \tag{39.6}$$

It is interesting to note that phone models are used only for decoding, and not for scoring when deter-

mining the identified language. We can also expand the phone model collection to include multiple sets of models, each trained by speech examples from a subset of languages. Now the set of F phone models, $\{\lambda_1^{\mathrm{AM}}, \ldots, \lambda_f^{\mathrm{AM}}, \ldots, \lambda_F^{\mathrm{AM}}\}$, are all used to decode a given spoken utterance, resulting in a set of F phone sequences, $Q = \{q_1, \ldots, q_f, \ldots, q_F\}$. Furthermore for the l-th language, we can train F sets of phone language models, $\{\lambda_{l,1}^{\mathrm{LM}}, \ldots, \lambda_{l,f}^{\mathrm{LM}}, \ldots, \lambda_{l,F}^{\mathrm{LM}}\}$, to compute F language-specific scores, $\{P(q_f|\lambda_{l,f}^{\mathrm{LM}}), f = 1, \ldots, F\}$. These scores can then be combined to make language classification decisions. This the basic idea behind parallel PRLM (P-PRLM) [39.18], which is by far the most successful approach to LID.

39.4 Vector-Space Characterization (VSC)

Vector-space modeling has become a standard tool in IR systems since its introduction several decades ago [39.5]. It uses a vector to represent a text document or a query. It has also been applied to text categorization [39.41] in which training vectors are used to design a collection of topic classifiers. A vector-based TC approach to LID has been proposed recently [39.37].

Suppose that an utterance X, represented by a sequence of speech feature vectors O, is decoded or tokenized, into a *spoken document*, $d(X)$, consisting of a series of I acoustic units, $d(X) = \{t_1 \ldots, t_i \ldots, t_I\}$, where each unit is drawn from a universal inventory, $U = \{u_1, \ldots, u_j, \ldots u_J\}$, of J acoustic letters shared by all the spoken languages to be considered, such that $t_i \in U$. We are then able to establish a collection of acoustic words by grouping units occurring consecutively to obtain a vocabulary of M distinct words, $W = \{w_1, \ldots, w_m, \ldots w_M\}$, such that each w_m can be a single-letter word like (u_j), a double-letter word like $(u_j u_k)$, a triple-letter word like $(u_j u_k u_l)$, and so on. Usually the vocabulary size, M, is equal to the total number of n-gram patterns needed to form words, e.g., $M = J + J \times J + J \times J \times J$ if acoustic words up to three tokens in length are considered valid. Next we can use some form of function $f(w_m)$, such as LSI [39.33], to evaluate the significance of having the word w_m in the document, $d(X)$. We are now ready to establish an M-dimension feature vector, $\boldsymbol{v} = [f(w_1), \ldots, f(w_m), \ldots f(w_M)]^{\mathrm{T}}$ in which $\boldsymbol{x}^{\mathrm{T}}$ denotes the transpose of the vector $\boldsymbol{x}$ for each spoken document.

It is clear that we need a number of fundamental units sufficient to cover the acoustic variation in the sound space. However a large J will result in a feature vector with a very high dimension if we would like to cover as many unit combinations when forming acoustic words. For example, with a moderate value of $J = 256$, we have $M = 65\,792$ even when we only consider words less than or equal to two letters. This is already a very large dimensionality not commonly utilized in speech and language processing algorithms. Finally, a vector-based classifier evaluates a goodness of fit, or score function $S_l(\boldsymbol{v}) = S(\boldsymbol{v}|\lambda_l)$ between a given vector $\boldsymbol{v}$ and a model of the l-th spoken language λ_l to make a decision. Any vector-based classifier can be used to design spoken language identification and verification systems. A goodness of fit, such as an inner product using a linear discriminant function (LDF) [39.42], can be used to evaluate vector-based scores:

$$S(\boldsymbol{v}|\lambda_l) \propto \gamma_l^{\mathrm{T}} \cdot \boldsymbol{v}\,, \tag{39.7}$$

where γ_l is a language-dependent weight vector of equal dimensionality to $\boldsymbol{v}$, with each attribute representing the contribution of its individual n-gram probability to the overall language score. The spoken document vector, $\boldsymbol{v}$ in (39.7), is high dimensional in nature when patterns up to triplets are included. When multiple inventories of models, like in P-PRLM, are used to produce a collection of F document vectors, $\{\boldsymbol{v}_f, f = 1, \ldots, F\}$, a large composite document vector, or supervector, $\boldsymbol{v} = [\boldsymbol{v}_1^{\mathrm{T}}, \ldots, \boldsymbol{v}_f^{\mathrm{T}}, \ldots, \boldsymbol{v}_F^{\mathrm{T}}]^{\mathrm{T}}$ can be established by stacking these F vectors. It has been shown that such supervectors have more discrimination power than single document vectors for language recognition [39.35].

Term weighting [39.43] is widely used to render the value of the attributes in a document vector by taking into account the frequency of occurrence of each attribute. It is interesting to note that attribute patterns that occur often in a few documents but not as often in others give high indexing power to these documents. On the other hand, patterns that occur very often in all documents possess little indexing power. This desirable property leads to a number of term weighting schemes, such as *tf-idf* (term-frequency-inverse document frequency) [39.44] and LSI [39.33], which are common for information retrieval [39.5], natural language call routing [39.45], and text categorization [39.41]. VSC is motivated by the same ideas, which aim to derive weights that dis-

criminate between languages using high-dimensional salient-feature vectors.

After representing a spoken document as a vector of statistics of AWs, LID becomes a vector-based classification problem. Many classifier designs in the machine learning literature for high-dimensional vector classification are readily available. For example conventional techniques, such as SVM [39.39], classification and regression tree (CART) [39.46] and artificial neural networks (ANN) [39.47], can be used to train vector-based classifiers, such that LID can be solved in the form:

$$\hat{l} = \arg\max_{l} S(\boldsymbol{v}|\lambda_l) \,. \tag{39.8}$$

We can also consider an indirect vectorization mechanism based on the scores computed for all the classes of interest, obtained from an ensemble of classifiers. Since these scores characterize the distribution of the outputs of a variety of classifiers, they do provider significant discriminatory power among classes, and can be grouped together to form *score supervectors* describing the overall behavior of a score distribution over different classifiers for all competing classes. One example of such a supervector is to concatenate HMM state scores from all the competing models to form an overall score vector. This approach has been shown to have good discriminatory power in isolated letter recognition [39.48, 49]. Since these score supervectors are usually obtained from a finite set of ensemble classifiers, their dimensionality is often much lower than that of the spoken document vectors discussed above. These supervectors are more amendable to treatment using conventional probabilistic modeling frameworks.

39.5 Spoken Language Verification

So far we have only discussed spoken language identification. Another recognition problem of interest is spoken language verification, which can be cast as a statistical hypothesis testing problem, i. e., testing a null hypothesis H_0 that an utterance O is from a claimed language against an alternative hypothesis H_1 that O is not from that language. Many issues we will discuss in this section have similarities to those commonly addressed in speaker and utterance verification [39.50]. According to the Neyman–Pearson lemma [39.51], an optimal test can be formulated as: given a test speech O, accept H_0 if

$$\frac{P(O|H_0)}{P(O|H_1)} > r_{\text{th}} \,, \tag{39.9}$$

where $P(O|H_0)$ and $P(O|H_1)$ are the probability distributions for the null and alternative hypotheses, respectively. The constant r_{th} is a verification threshold. The test in (39.9) is known as a probability ratio test. The ratio $P(O|H_0)/P(O|H_1)$ is called a *probability ratio statistic*. The threshold r_{th} is referred to as the *critical value*; the region $A = \{O : [P(O|H_0)/P(O|H_1)] > r_{\text{th}}\}$ is called the acceptance region, and the region $\overline{A}$, containing points not in A, is called the critical region of the test.

There are two verification decisions to be made, namely rejection and acceptance of the null hypothesis. Correspondingly there are two types of errors, namely false rejection (type I) and false acceptance (type II). The power of a test is defined as the probability of correct rejection, i. e., rejecting H_0 when it should be rejected, which is one minus the maximum of the probability of a type II error. The level of significance of a test is defined as the maximum of the probability of type I errors. The level of significance is computed based on the distribution of the likelihood test statistic and the choice of the threshold.

To test some simple hypotheses under a few regularity conditions, a generalization of the Neyman–Pearson lemma shows that a likelihood-ratio test (LRT) is the *most powerful test* (smallest type II error test) for a given level of significance (type I error) if $P(O|H_0)$ and $P(O|H_1)$ are known exactly. For testing more-complex composite hypotheses, an optimal test is usually hard to come by. Furthermore in most practical verification applications we do not have access to complete knowledge of H_0 and H_1, and therefore optimal tests cannot be designed. In these cases we can attempt to evaluate $P(O|H_0)$ and $P(O|H_1)$ based on some assumed forms for their distributions, such as those discussed in Sect. 39.2, in which case a probability ratio test can still be used.

Even in cases where a probabilistic characterization is not easy to determine, we can still compute scores such as $S(O|H_0)$ and $S(O|H_1)$, so that the VSC models presented in Sect. 39.3 are equally applicable to the design of spoken language verification systems. Even with probabilistic modeling we can also evaluate scores of the form $S(O|H_i) = \log P(O|H_i)$ so we can use any score to compute a generalized log likelihood ratio (GLLR) test

statistic, which can be considered as a distance measure:

$$d(O, \Lambda) = -S(O|\lambda_{H_0}) + S(O|\lambda_{H_1}) , \qquad (39.10)$$

where $\Lambda = (\lambda_{H_0}, \lambda_{H_1})$ is the collection of all models. We then accept H_0 if $d(O, \Lambda) < \tau_{\text{th}}$. Here τ_{th} is another form of the verification threshold. This formulation can be applied directly to two-class verification problems. However the alternative hypothesis H_1 is often difficult to model. For example if H_0 is defined for a claimed language l, H_1 is often a composite hypothesis representing all other competing languages other than l. There are often more negative and diverse samples to train the *impostor* model λ_{H_1} than positive and consistent samples to train the target model λ_{H_0}.

One way to alleviate this difficulty is to train one model for each of the languages of interest to obtain a collection of models $\Lambda = (\lambda_1, \ldots, \lambda_L)$, one for each language. Now, to verify whether an unknown utterance O is generated from a claimed language l, the target score can be computed as $S_l(O|H_0) = S(O|\lambda_l)$. Furthermore we can assume that $S_l(O|H_1)$ is a function of all the competing language scores other than $S(O|\lambda_l)$. One approach is to assume that $S_l(O|H_1)$ is evaluated as an antidiscriminant function commonly used in discriminative training (DT) in ASR [39.52,53] to compute a geometric average of all competing scores as

$$S_l(O|H_1) = \log \left\{ \frac{1}{L-1} \sum_{i \neq l} \exp[\eta S(O|\lambda_i)] \right\}^{\frac{1}{\eta}} , \qquad (39.11)$$

where η is a positive constant. It is noted that the right-hand side of (39.11) is the L_η norm in real analysis, and that it converges to $\max_i S(O|\lambda_i)$ as $\eta \to \infty$. This score has been used in multicategory (MC) text categorization [39.37], and it was shown that MC TC outperforms binary TC, especially in cases when there are very few positive examples, sometimes only one sample, to train a topic classifier. The same idea was also applied to the design of language verification systems in the 2005 NIST language recognition evaluation [39.42]. Part of the successes in [39.37] and [39.42] can be attributed to discriminative classifier learning, which will be discussed in Sect. 39.6.

By now it is clear that the modeling and classifier design techniques discussed in Sects. 39.3 and 39.4 for language identification can be extended to language verification as well. One remaining key issue is the selection of verification thresholds, which is critical in designing high-performance spoken language verification systems for real-world applications. They are often determined empirically according to the specific application requirements. Since the distance measure in (39.10) also characterizes a separation between the target and competing models [39.54], we can plot histograms of $d_l(O, \Lambda)$ over positive and negative training samples, respectively. The size of the overlap region of the two curves is often a good indicator for predicting type I and type II errors. The verification thresholds can then be determined by selecting a value in the overlap region to balance the false-rejection and false-acceptance errors. One good illustration was demonstrated in the recent 2005 NIST LRE [39.42].

39.6 Discriminative Classifier Design

We have now briefly addressed most of the research issues in designing language recognition systems. More detail can be found in the last two chapters of this Part G. Next we will discuss an important methodology for discriminative training that has created a lot of enthusiasm in the fields of speech recognition [39.52,53], utterance verification [39.55], and text categorization [39.41], but not yet been fully utilized in the language recognition community. So far SVM is still the dominating discriminative classifier design approach to LID. However the techniques used in SVM training are not easily extended to other popular classifiers, such as GMM, HMM, ANN, LDF, and CART. In the following we describe a general framework to formulate a broad family of DT algorithms for any classifier based on any performance metric.

Consider a set of training patterns $\boldsymbol{X} = \{X_i, 1 \leq i \leq N\}$, with N being the total number of training tokens, and O_i, an observation vector sequence associated with X_i. Each token belongs to one of L classes, C_l, $1 \leq l \leq L$. The goal of pattern classification is to use the labeled set X to design a decision rule based on a set of parameters $\Lambda = \{\lambda_l, l = 1, ..., L\}$ such that the classification error is minimized. The optimal classification approach is the Bayes decision rule if the a priori probabilities $P(C_l)$ and the class-conditional probability $P(O|C_l)$ are known. Unfortunately, in pattern recognition applications we rarely have complete knowledge of the probabilistic structure of the problem. We only have some vague knowledge about the distribution and are given a finite number of training samples to design the

classifiers. A conventional approach to this problem is to assume a parametric form for the conditional densities and obtain estimates of the parameters from the given finite set of labeled training patterns. However, any assumed parametric distributions will create some mismatch between the true and estimated conditional distributions. The limited availability of training data is another problem for effective characterization of the conditional densities.

An alternative approach that alleviates these problems is to use a set of class discriminant functions (similar to the score functions discussed above) $\{g_l(O, \Lambda) : l = 1, \ldots, L\}$ to perform classification and hypothesis testing, and to use discriminative training to estimate the parameters Λ. The form of the probability distributions is not needed in this case. Each $g_l(O; \Lambda)$ evaluates the similarity between a given test utterance O and the class C_l. To combine current speech pattern classification techniques and the discriminative training approach we can use the conditional class distributions as the class discriminant functions and obtain the classifier parameters accordingly based on minimum error classification (MCE) [39.52,53] and minimum verification error (MVE) [39.55] training. Recently maximal figure-of-merit (MFoM) [39.41] learning for vector-based classifiers has also been shown to give superior and robust performance compared with SVM-trained classifiers in text categorization.

Three additional functions are required to formulate discriminative training, namely:

1. the class antidiscriminant function, $G_l(O; \Lambda)$ [similar to the quantity in (39.11)]
2. the class misclassification measure, $d_l(O; \Lambda)$, which is usually defined as the difference between $G_l(O; \Lambda)$ and $g_l(O; \Lambda)$ [similar to the measure in (39.10)]
3. the class loss function, $l_l(O; \Lambda) = l[d_l(O; \Lambda)]$, which measures the loss or cost incurred for a given value of $d_k(O; \Lambda)$

The loss is often a smooth 0–1 function such as a sigmoid to approximate error counts based on its distance from the decision boundary, $d_l(O; \Lambda) = \beta_l$:

$$l_l(O; \Lambda) = \frac{1}{1 + \exp[-\alpha_l(d_l - \beta_l)]} , \tag{39.12}$$

where α_l and β_l are parameters for the sigmoid function characterizing the slope near, and the location of, the decision boundary, respectively.

Given the four sets of functions the optimization objective functions can be defined differently, depending on the performance metrics to be optimized in each application. For language identification, we are interested in minimizing the overall approximate empirical classification error rate for all training data, $\boldsymbol{O} = \{O_i, 1 \le i \le N\}$:

$$L(\boldsymbol{O}, \Lambda) = \frac{1}{N} \sum_{i=1}^{N} \sum_{l=1}^{L} l(d_l) \mathbf{1}(O_i \in C_l) , \tag{39.13}$$

where the function $\mathbf{1}(S)$ is the indicator function for a logical variable S. For language verification we can approximate the empirical false-rejection (type I) and false-acceptance (type II) error rates for the l-th language as

$$L_{l1}(\boldsymbol{O}, \Lambda) = \frac{1}{N_l} \sum_{i=1}^{N_l} l(d_l) \mathbf{1}(O_i \in C_l) \tag{39.14}$$

and

$$L_{l2}(\boldsymbol{O}, \Lambda) = \frac{1}{\overline{N}_l} \sum_{i=1}^{\overline{N}_l} l(d_l) \mathbf{1}(O_i \in \overline{C}_l) , \tag{39.15}$$

where N_l and $\overline{N}_l$ are the numbers of tokens in and not in class C_l in the training set, respectively, and $\overline{C}_l$ denotes the set of all tokens not belonging to the class C_l. According to the application requirements we can specify ω_{l1} and ω_{l2}, for the l-th class, as the costs of making type I and II errors. Now the overall empirical average cost can be defined as:

$$L(\boldsymbol{O}, \Lambda) = \frac{1}{L} \sum_{l=1}^{L} \omega_{l1} L_{ll}(\boldsymbol{O}, \Lambda) + \omega_{l2} L_{l2}(\boldsymbol{O}, \Lambda) . \tag{39.16}$$

In IR and TC applications precision, *recall*, and $F1$ measures are commonly adopted as performance metrics. They are defined for the l-th class as:

$$\mathrm{Pr}_l(\boldsymbol{O}, \Lambda) = \frac{(1 - L_{l1}) \times N_l}{(1 - L_{l1}) \times N_l + L_{l2} \times \overline{N}_l} , \tag{39.17}$$

$$\mathrm{Re}_l(\boldsymbol{O}, \Lambda) = 1 - L_{l1} , \tag{39.18}$$

and

$$\mathrm{F1}_l(\boldsymbol{O}, \Lambda) = \frac{2 \times \mathrm{Pr}_l \times \mathrm{Re}_l}{\mathrm{Pr}_l + \mathrm{Re}_l} . \tag{39.19}$$

Now we can define the overall loss objective as the negative of the average F1 value over all L classes:

$$L(\boldsymbol{O}, \Lambda) = -\frac{1}{L} \sum_{l=1}^{L} \mathrm{F1}_l(\boldsymbol{O}, \Lambda) . \tag{39.20}$$

The three empirical loss functions in (39.13), (39.16), and (39.20) are the objectives to be minimized in MCE, MVE, and MFoM discriminative learning, respectively. Generalized probabilistic descent (GPD) algorithms [39.53] are often used to solve these optimization problems such that Λ is adjusted from one iteration to the next according to

$$\Lambda_{q+1} = \Lambda_q + \Delta\Lambda_q \tag{39.21}$$

with Λ_q being the parameter set at the q-th iteration. The correction term $\Delta\Lambda_q$, in a batch mode, is:

$$\Delta\Lambda_q = -\epsilon_q V_q \nabla L(\Lambda_q)\,, \tag{39.22}$$

where V_q is a positive-definite learning matrix and ϵ_q is a small positive real number representing the learning step size. We usually set V_q to be a diagonal matrix and define $\epsilon_q = (1 - q/Q_c)$, with Q_c denoting a prescribed maximum number of iterations to meet the convergence conditions, as required by the stochastic gradient search algorithm. GPD algorithms usually converge slowly to local minima. To speed up their convergence, QuickProp algorithms can also be applied to adjust the learning rate dynamically [39.56]. Globally optimal algorithms are also being actively pursued by the machine learning community.

These mentioned discriminative training algorithms are flexible enough to deal with any performance metric for any given classifier as long as these metrics can be expressed as functions of the classifier parameters. They can also be considered as a decision feedback mechanism in that every training token will go through a classification process first and its contributions to parameter adjustment depend on how well the decision is made. Therefore they usually work very well on the training data. On the other hand, SVM-based learning algorithms are designed to provide a *margin* to serve as a tolerance region around the decision boundaries implied by the classifiers. In so doing they usually provide better generalization capabilities than other learning algorithms. Since the test risk is often expressed as a function of the empirical risk and a regularization penalty term as a function of the VC dimension [39.39], we can design a family of margin-based discriminative training algorithm to enhance both the accuracy and robustness of the classifiers. This is a promising new direction, and some attempts in this direction have recently been proposed for estimating HMM parameters (e.g., [39.57]).

39.7 Summary

In this chapter, we have provided an overview of the three chapters to be presented later in this Part G. We have also highlighted two currently popular approaches to spoken language identification, namely phone recognition followed by language modeling and vector-space characterization. We demonstrated that the techniques developed for LID can be directly extended to spoken language verification if we can properly evaluate scores for the competing null and alternative hypotheses. Finally, we briefly presented a general framework for discriminative classifier design of non-SVM classifiers. The field of spoken language classification has witnessed rapid technological progress in recent years. We expect this trend to continue as novel paradigms are explored and high-performance systems are developed.

References

39.1 L.R. Rabiner, B.-H. Juang: *Fundamentals of Speech Recognition* (Prentice Hall, Englewood Cliffs 1993)

39.2 W.M. Campbell, D.E. Sturim, D.A. Reynolds: Support vector machines using GMM supervectors for speaker recognition, IEEE Signal Process. Lett. **13**(5), 308–311 (2006)

39.3 W.M. Campbell, J.P. Campbell, D.A. Reynolds, E. Singer, P.A. Torres-Carrasquillo: Support vector machines for speaker and language recognition, Comput. Speech Lang. **20**(2-3), 210–229 (2005)

39.4 H. Li, B. Ma, C.-H. Lee: A vector space modeling approach to spoken language identification, IEEE Trans. Audio Speech Lang. Process. **15**(1), 271–284 (2007)

39.5 G. Salton: *The SMART Retrieval System* (Prentice-Hall, Englewood Cliffs 1971)

39.6 F. Sebastiani: Machine learning in automated text categorization, ACM Comput. Surv. **34**(1), 1–47 (2002)

39.7 T. Joachims: *Learning to Classify Text Using Support Vector Machines* (Kluwer Academic, Dordrecht 2002)

39.8 C.-H. Lee: From knowledge-ignorant to knowledge-rich modeling: a new speech research paradigm for next generation automatic speech recognition, Proc. ICSLP (2004) pp. 109–112

39.9 V.W. Zue, J.R. Glass: Conversational interfaces: advances and challenges, Proc. IEEE **88**(8), 1166–1180 (2000)

39.10 A. Waibel, P. Geutner, L.M. Tomokiyo, T. Schultz, M. Woszczyna: Multilinguality in speech and spoken language systems, Proc. IEEE **88**(8), 1181–1190 (2000)

39.11 B. Ma, C. Guan, H. Li, C.-H. Lee: Multilingual speech recognition with language identification, Proc. ICSLP (2002) pp. 505–508

39.12 P. Dai, U. Iurgel, G. Rigoll: A novel feature combination approach for spoken document classification with support vector machines, Proc. Multimedia Information Retrieval Workshop (2003) pp. 1–5

39.13 K. Kirchhoff, S. Parandekar, J. Bilmes: Mixed memory Markov models for automatic language identification, Proc. ICASSP (2002) pp. 761–764

39.14 M. Sugiyama: Automatic language recognition using acoustic features, Proc. ICASSP (1991) pp. 813–816

39.15 A.G. Adami, H. Hermansky: Segmentation of speech for speaker and language recognition, Proc. Eurospeech (2003) pp. 841–844

39.16 M. Adda-Decker, F. Antoine, P.B. de Mareuil, I. Vasilescu, L. Lamel, J. Vaissiere, E. Geoffrois, J.-S. Liénard: Phonetic knowledge, phonotactics and perceptual validation for automatic language identification, Proc. ICPhS (2003) pp. 747–750

39.17 T.J. Hazen: *Automatic Language Identification Using a Segment-Based Approach*, M.Sc. Thesis (MIT, New York 1993)

39.18 M.A. Zissman: Comparison of four approaches to automatic language identification of telephone speech, IEEE Trans. Speech Audio Process. **4**(1), 31–44 (1996)

39.19 D. Matrouf, M. Adda-Decker, L.F. Lamel, J.-L. Gauvain: Language identification incorporating lexical information, Proc. ICSLP (1998) pp. 181–184

39.20 P.A. Torres-Carassquilo, E. Singer, M.A. Kohler, R.J. Greene, D.A. Reynolds, J.R. Deller Jr.: Approaches to language identification using Gaussian mixture models and shifted delta cepstral features, Proc. ICSLP (2002) pp. 89–92

39.21 Y.K. Muthusamy, N. Jain, R.A. Cole: Perceptual benchmarks for automatic language identification, Proc. ICASSP (1994) pp. 333–336

39.22 C.-H. Lee, F.K. Soong, K.K. Paliwal (Eds.): *Automatic Speech and Speaker Recognition: Advanced Topics* (Kluwer Academic, Dordrecht 1996)

39.23 C.-H. Lee, Q. Huo: On adaptive decision rules and decision parameter adaptation for automatic speech recognition, Proc. IEEE **88**(8), 1241–1269 (2000)

39.24 J.L. Gauvain, L. Lamel: Large-vocabulary continuous speech recognition: advances and applications, Proc. IEEE **88**(8), 1181–1200 (2000)

39.25 L.R. Rabiner: A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE **77**(2), 257–286 (1989)

39.26 H. Li, B. Ma: A phonotactic language model for spoken language identification, Proc. ACL (2005) pp. 515–522

39.27 E. Singer, P.A. Torres-Carrasquillo, T.P. Gleason, W.M. Campbell, D.A. Reynolds: Acoustic, phonetic and discriminative approaches to automatic language recognition, Proc. Eurospeech (2003) pp. 1345–1348

39.28 Y. Yan, E. Barnard: An approach to automatic language identification based on language dependent phone recognition, Proc. ICASSP (1995) pp. 3511–3514

39.29 K.M. Berkling, E. Barnard: Language identification of six languages based on a common set of broad phonemes, Proc. ICSLP (1994) pp. 1891–1894

39.30 T. Nagarajan, H.A. Murthy: Language identification using parallel syllable-like unit recognition, Proc. ICASSP (2004) pp. 401–404

39.31 K.M. Berkling, E. Barnard: Analysis of phoneme-based features for language identification, Proc. ICASSP (1994) pp. 289–292

39.32 P.A. Torres-Carrasquillo, D.A. Reynolds, R.J. Deller Jr.: Language identification using Gaussian mixture model tokenization, Proc. ICASSP (2002) pp. 757–760

39.33 J.R. Bellegarda: Exploiting latent semantic information in statistical language modeling, Proc. IEEE **88**(8), 1279–1296 (2000)

39.34 C.E. Shannon: Prediction the Entropy of Printed English, Bell Syst. Tech. J. **30**, 50–64 (1951)

39.35 H. Li, B. Ma, R. Tong: Vector-based spoken language recognition using output coding, Proc. Interspeech (2006)

39.36 C.-H. Lee, F.K. Soong, B.-H. Juang: A segment model based approach to speech recognition, Proc. ICASSP (1988) pp. 501–504

39.37 S. Gao, B. Ma, H. Li, C.-H. Lee: A text-categorization approach to spoken language identification, Proc. Interspeech (2005) pp. 2837–2840

39.38 D.A. Reynolds, R.C. Rose: Robust text-independent speaker identification using Gaussian mixture speaker models, IEEE Trans. Speech Audio Process. **3**(1), 72–83 (1995)

39.39 V. Vapnik: *The Nature of Statistical Learning Theory* (Springer, Berlin, Heidelberg 1995)

39.40 W.M. Campbell, T. Gleason, J. Navratil, D. Reynolds, W. Shen, E. Singer, P.A. Torres-Carrasquillo: Advanced language recognition using cepstra and phonotactics: MITLL system performance on the NIST 2005 language recongnition evaluation, Proc. IEEE Odyssey Speaker and Language Recognition Workshop (2006)

39.41 S. Gao, W. Wu, C.-H. Lee, T.-S. Chua: A MFoM learning approach to robust multiclass multi-label text categorization, Proc. ICML (2004) pp. 42–49

39.42 J. Li, S. Yaman, C.-H. Lee, B. Ma, R. Tong, D. Zhu, H. Li: Language recognition based on score distribution feature vectors and discriminative classfier fusion, Proc. IEEE Odyssey Speaker and Language Reognition Workshop (2006)

39.43 K.S. Jones: A statistical interpretation of term specificity and its application in retrieval, J. Docum. **28**, 11–20 (1972)

39.44 J. Chu-Carroll, B. Carpenter: Vector-based natural languagecall routing, Computat. Linguist. **25**(3), 361–388 (1999)

39.45 H.K.J. Kuo, C.-H. Lee: Discriminative training of natural language call routers, IEEE Trans. Speech Audio Process. **11**(1), 24–35 (2003)

39.46 L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone: *Classification and Regression Trees* (Chapman Hall, New York 1984)

39.47 S. Haykin: *Neural Networks: A Comprehensive Foundation* (McMillan, Englewood 1994)

39.48 S. Katagiri, C.-H. Lee: A new hybrid algorithm for speech recognition based on HMM segmentation and discriminative classification, IEEE Trans. Speech Audio Process. **1**(4), 421–430 (1993)

39.49 K.-Y. Su, C.-H. Lee: Speech recognition using weighted HMM and subspace projection approaches, IEEE Trans. Speech Audio Process. **2**(1), 69–79 (1994)

39.50 C.-H. Lee: A unified statistical hypothesis testing approach to speaker verification and verbal information verification, Proc. COST Workshop on Speech Technology in the Public Telephone Network: Where are we today? (1997) pp. 62–73

39.51 E.L. Lehmann: *Testing Statistical Hypotheses* (Wiley, New York 1959)

39.52 B.-H. Juang, W. Chou, C.-H. Lee: Discriminative methods for speech recognition, IEEE Trans. Speech Audio Process. **5**(3), 257–265 (1997)

39.53 S. Katagiri, B.-H. Juang, C.-H. Lee: Pattern recognition using a generalized probabilistic descent method, Proc. IEEE **86**(11), 2345–2373 (1998)

39.54 Y. Tsao, J. Li, C.-H. Lee: A study on separation between acoustic models and its applications, Proc. InterSpeech (2005)

39.55 M. Rahim, C.-H. Lee: String-based minimum verification error (SB-MVE) training for speech recognition, Comput. Speech Lang. **11**(2), 147–160 (1997)

39.56 S.E. Fahlman: An empirical study of learning speed in back-propagation networks, CMU CS Tech. Rep. **CMU-CS-88-162** (1998)

39.57 J. Li, M. Yuan, C.-H. Lee: Soft margin estimation of hidden Markov model parameters, Proc. InterSpeech (2006)

40. Spoken Language Characterization

M. P. Harper, M. Maxwell

This chapter describes the types of information that can be used to characterize spoken languages. Automatic spoken language identification (LID) systems, which are tasked with determining the identity of the language of speech samples, can utilize a variety of information sources in order to distinguish among languages. In this chapter, we first define what we mean by a language (as opposed to a dialect). We then describe some of the language collections that have been used to investigate spoken language identification, followed by discussion of the types of features that have been or could be utilized by automatic systems and people. In general, approaches used by people and machines differ, perhaps sufficiently to suggest building a partnership between human and machine. We finish with a discussion of the conditions under which textual materials could be used to augment our ability to characterize a spoken language.

As we move into an increasingly globalized society, we are faced with an ever-growing need to cope with a variety of languages in computer-encoded text documents, documents in print, hand-written (block and cursive) documents, speech recordings of various qualities, and video recordings potentially containing both speech and textual components. A first step in coping with these language artifacts is to identify their language (or languages).

The scope of the problem is daunting given that there are around 7000 languages spoken across the world, as classified by the Ethnologue [40.1]. Indeed it can be difficult to collect materials for all of these languages, let alone develop approaches capable of discriminating among them. Yet the applications that would be supported by the ability to effectively and efficiently identify the language of a text or speech input are compelling: document (speech and text) retrieval, automated routing to machine translation or speech recognition systems, spoken dialog systems (e.g., for making travel arrangements), data mining systems, and systems to route emergency calls to an appropriate language expert.

In practice, the number of languages that one might need to identify is for most purposes much less than 7000. According to the Ethnologue, only about 330 languages have more than a million speakers. Thus, one might argue that in practice, there is a need for language identification (ID) of a set of languages that number perhaps in the hundreds. For example, Language Line Services [40.2] provides interpreter services to public and private clients for 156 languages, which they claim represents around 98.6% of all their customer requests for language services. Of course, for some purposes, the set of languages of interest could be far smaller.

This chapter discusses the knowledge sources that could be utilized to characterize a spoken language in order to distinguish it automatically from other languages. As a first step, we define what we mean by a language (as opposed to a dialect). We then describe some of the language collections that have been used to investigate spoken language identification, followed by a discussion of the types of linguistic features that have been or could be used by a spoken language identification (LID) system to determine the identity of the language of a speech sample. We also describe the cues that humans can use for language identification. In general, the approaches used by people and machines differ, perhaps sufficiently to point the way towards building a part-

nership between human and machine. We finish with a discussion of the conditions under which textual materials could be used to augment our ability to characterize a spoken language.

40.1 Language versus Dialect

When talking about language identification, it is important to define what we mean by a language. At first glance, the definition of *language* would seem to be simple, but it is not. Traditionally, linguists distinguish between languages and dialects by saying that dialects are mutually intelligible, unlike distinct languages. But in practice, this distinction is often unclear: mutual intelligibility is relative, depending on the speaker's and listener's desire to communicate, the topic of communication (with common concepts being typically easier to comprehend than technical concepts or concepts which happen to be foreign to one or the other participant's culture), familiarity of the hearer with the speaker's language (with time, other *accents* become more intelligible), degree of bilingualism, etc.

Political factors can further blur the distinction between dialect and language, particularly when two peoples want – or do not want – to be considered distinct. The questionable status of Serbo-Croatian is an obvious example; this was until recently considered to be more or less a single unified language. But with the breakup of Yugoslavia, the Serbian and Croatian languages (and often Bosnian) have been distinguished by many observers, despite the fact that they are largely mutually intelligible. The status of the various spoken varieties of Arabic is an example of the opposite trend. While many of these varieties are clearly distinct languages from the standpoint of mutual intelligibility, the desire for Arab unity has made some claim that they are merely dialects [40.3, 4].

Writing systems may also cause confusion for the distinction between languages and dialects. This is the case for Hindi and Urdu, for example, which in their spoken form are for the most part mutually intelligible (differing slightly in vocabulary). But they use radically different writing systems (Devanagari for Hindi, and a Perso-Arabic script for Urdu), in addition to being used in different countries, and so are generally treated as two different languages [40.5]. Hence, the distinction between language and dialect is often unclear, and so in general it is better viewed as a continuum rather than a dichotomy.

At a higher level, one can characterize a language by its language family, which is a phylogenetic unit such that all members are descended from a common ancestor (languages that cannot be reliably classified into a family are called language isolates). For example, Romance languages (e.g., French, Spanish), Germanic languages (e.g., German, Norwegian), Indo-Aryan languages (e.g., Hindi, Bengali), Slavic languages (e.g., Russian, Czech), and Celtic languages (e.g., Irish and Scots Gaelic) among others are believed to be descended from a common ancestor language some thousands of years ago, and hence are grouped into the Indo-European language family. Language families are often subdivided into branches, although the term family is not restricted to one level of a language tree (e.g., the Germanic branch of Indo-European language family is often called the Germanic family). As a result of their common descent, the languages of a language family or branch generally share characteristics of phonology, vocabulary, and grammar, although these shared resemblances may be obscured by changes in a particular language, including borrowings from languages of other language families. Thus, while English is a Germanic language, its grammar is substantially different from that of other Germanic languages, and a large portion of its vocabulary is derived from non-Germanic languages, particularly French [40.6].

The issues of language, dialect, and language family have repercussions for systems that try to assign a language tag to a piece of text or a sample of speech, and in particular for the International Organization for Standardization (ISO) standard language codes. The original standard, ISO 639-1, listed only 136 codes to distinguish languages. This allowed for the identification of most *major* languages, but not minor languages. In fact some of the codes did not represent languages at all. The code for *Quechua*, for example, corresponds to an entire language family consisting of a number of mutually unintelligible languages; arguably worse, the code for *North American Indian* refers to a number of language families, most of which include several distinct languages. For an analysis of some of the problems with ISO 639-1 and its revision, ISO 639-2, see [40.7].

At the other end of the spectrum of language classification is the Ethnologue [40.1], a listing of nearly 7000 languages. This work takes an explicitly linguistic

view of language classification, i. e., it attempts to assign distinct names to all and only mutually unintelligible varieties. Many observers have accused the Ethnologue of being a *splitter*, i. e., of claiming too many distinctions. This is a debatable point, but it does serve to highlight a fundamental problem of classifying a language sample as belonging to this language or that: it is hard to classify a language artifact if we cannot agree ahead of time what the possibilities are. And indeed, for some purposes, the finer-grained classification like the Ethnologue may be superfluous, while for others it may be crucial. One example of the need for fine-grained classification would be Arabic *dialect* identification, where it may be desirable to determine which variety of Arabic someone is speaking.

To some extent, the problem of conflicting classification criteria (ISO versus Ethnologue) has been resolved by unifying the two systems as ISO 639-3 [40.8]. In addition to the set of 7000 languages already listed in the Ethnologue, ISO 639-3 adds various other languages, including extinct languages and artificial languages (such as Klingon) and so-called *macro languages*, which are really language families (such as Arabic). However, even given that the standard can now be agreed on – the set of languages in ISO 639-3 – there does remain a difficulty for language identification, namely the level of granularity for language classification. The reader should remember that the problem exists, and that identifying a text or speech recording as *Arabic* may be adequate for some purposes, but insufficient for others.

The continuum from language to dialect has another implication for language identification: the existence of significant differences among dialects of a single language can make identification of that language more difficult, particularly in its spoken form (written forms of languages tend to be more standardized [40.9]). Dialectal differences can occur in all linguistic aspects: lexicon, grammar (syntax and morphology), and phonology; but it will probably be the phonology that causes the most problems for language identification in speech, since this is the level of representation that is commonly used in existing systems.

In English, for example, the differences between rhotic and non-rhotic dialects are well known [40.10]. The Scots dialects of English demonstrate even greater differences from other dialects of English in their vowel systems, where they have largely lost the distinction between so-called *long* and *short* vowels; and among the consonants, Scots English has retained the voiceless velar fricative and the voiceless labiovelar, both of which have merged with other phonemes in other dialects of English [40.11]. See the SCOTS project [40.12] for some recorded speech examples. Likewise, spoken Mandarin is strongly influenced by the native dialect spoken in a region, as well as other factors such as age [40.13].

Dialects that differ primarily in their phonology are often called *accents*, although this term is also used to refer to pronunciation by a non-native speaker of a language. In this regard, while foreign accents might be dismissed as irrelevant to language identification for some purposes, in the case of major languages, there may be significant communities of non-native speakers who use the major language as a trade language. For example, in India there are 21 (currently) official languages, but English is defined by the Constitution of India, as well as by later laws, as one of the languages of communication for the federal government (the other being Hindi). The result of this and other factors is that English is spoken non-natively by a large portion of the Indian population, and it has acquired a distinctly Indian pronunciation as a consequence [40.14]. Indian English lacks voiceless aspirated stops; what would be alveolar consonants in other varieties of English are often retroflexed; and the stress patterns are altered with the effect that vowels that would be reduced in other Englishes to schwa appear instead in their nonreduced forms.

English is used as a trade language or lingua franca in many other parts of the world as well, and these local versions of English are often significantly different from the English spoken in countries where it is a native language (see [40.15] for descriptions of some of these varieties). The same is true of French, Hausa, and many other languages that are used as trade languages (cf. [40.16] and the Ethnologue [40.1]). Such lingua franca varieties can differ significantly from *standard* varieties in ways which would likely impact language identification.

Additionally, there is a substantial amount of variability in the spoken realization of a particular language [40.17–19] due to a variety of factors, including:

- mispronunciation
- individual speaking style
- genre (e.g., conversations versus formal presentations)
- variations in speaking rate
- the speaker's psychological state
- the speaker's language repertoire
- the social and economic background of the speaker
- the speaker's first language (where the speaker is speaking in a second language)
- channel characteristics

This variability creates a challenge when identifying the language of a speech sample. A greater understanding of language, dialect, and accent is an important first step in developing an approach for language identification.

40.2 Spoken Language Collections

Engineering research in spoken language identification has been based on a very small sampling of languages from the 7000 identified by the Ethnologue. The first multilingual speech collection targeted for language identification system evaluation, the Oregon Graduate Institute (OGI) multilanguage corpus, was released in 1994 and consisted of spoken responses to prompts recorded over telephone lines by speakers of 12 languages (see [40.20–22]). The set was selected to contain both unrelated languages (e.g., German, Vietnamese, and Tamil), as well as more closely related languages (e.g., English and German). It covers languages with various types of prosodic phenomena that occur in spoken languages, such as tone (e.g., Mandarin and Vietnamese) and pitch accents (Japanese), as well as languages with various levels of complexity at the syllable level. See appendix A in [40.23] for a family language tree for the languages in this collection and appendix B for a table comparing the phone inventories for these languages. Perhaps one of the most important requirements for the languages selected was the ability to access sufficient numbers of speakers of the language in the United States, a factor that has been important in collections used for evaluation.

A second collection, the Linguistic Data Consortium (LDC) CallFriend corpus (see the list under the LID heading at [40.24]), was released in 1996 and contains telephone conversations among speakers of 14 languages or dialects, most of which appeared in the OGI multilanguage corpus. This collection was used in National Institute of Standards and Technology (NIST) evaluations of LID systems in 1996 and 2003. The CallHome collection, which was released between 1996 and 1997, is an additional multilingual resource containing six languages that were collected for large-vocabulary speech recognition; however, it does not expand the number of languages beyond what was available from CallFriend. Given the limited sampling of languages in these collections, it is not surprising that researchers developing spoken LID systems have not investigated whether languages naturally cluster into groups that parallel the classification of languages into families and branches.

Additional speech collections have been developed with an increased number of languages. The Center for Spoken Language Understanding (CSLU) 22 languages corpus, which was initially collected from 1994 through 1997, contains spoken utterances in 21 languages. It has gone through several versions, resulting in an increased number of transcribed utterances, and was released through LDC in 2005 (see [40.25]). LDC is also currently collecting the MIXER corpus which will contain telephone calls over 24 languages or dialects [40.26]. The 2005 NIST evaluation data contained speech from the Mixer and CallFriend corpora, as well as some data collected at the Oregon Health and Science University. The data in all of these collections will hopefully stimulate new methods for constructing speech-based LID systems. However, even 20 or so languages is far from the number of languages that would need to be identified to support Language Line Services.

The collection of significant quantities of comparable telephone speech in multiple languages has become more challenging in recent years. There is an implicit assumption that the training and development data are collected under conditions that are comparable to the evaluation materials. Data must also be collected in such a way that it is impossible to identify language based on speaker, gender distribution, domain, channel characteristics, etc. Finally, due to the falling cost of long-distance telephone calls, incentives such as free long-distance calls are less attractive to potential participants. This difficulty in collecting comparable speech corpora for a large number of languages has obvious implications for speech-based LID; we return to this issue later.

40.3 Spoken Language Characteristics

A speech-based LID model for a particular language is trained to represent the corresponding language and to differentiate it from others. Hence an important challenge to speech-based LID systems is the effective incorporation of discriminative knowledge sources into their models. Some systems use only the digitized

speech utterances and the corresponding true identities of the languages being spoken for training, whereas others require additional information such as phonetic and/or orthographic transcriptions, which can be expensive to produce. During the language recognition phase, a new audio sample is compared to each of the language-dependent models, and the language of the closest matching model (e.g., using maximum likelihood) is selected. We will discuss various levels of knowledge that can be used to identify a language in this section, touching upon research that utilizes the representation where appropriate.

Automatically Derived Features from the Speech Signal. Languages can be identified based on features that are automatically derived from the speech signal itself. Systems utilizing this type of information are motivated by the observation that different languages are made up of a variety of different sounds; hence, feature vectors automatically extracted from the speech signal over short time frames (segments) can be used to discriminate among the languages. Such systems typically use a multistep process (including modules to remove silence from the samples, to reduce channel effects, etc.) to convert the digitized speech signal into a feature vector representation. Given feature extraction and knowledge of which training samples correspond to each language, a classifier is constructed for each language based on language-dependent patterns of feature vectors. Methods include approaches that model only the static distribution of acoustic features given the language (e.g., [40.27]) and approaches that also utilize the patterns of change of these feature vectors over time (e.g., [40.28]). A variety of computational models have been investigated, including Gaussian mixture models [40.29], Hidden Markov models [40.29,30], artificial neural networks [40.22], and support vector machines [40.31]. Although these acoustic-based systems do not require training data that is labeled with explicit linguistic units such as phonemes or words, they also do not perform as accurately as systems that use linguistic knowledge; however, improved accuracy can be achieved by integrating acoustic- and linguistic-based knowledge sources [40.23, 32].

Phonological Information. Phonology is a branch of linguistics that studies sound systems of human languages [40.33]. In a given language, a phoneme is a symbolic unit at a particular level of representation; the phoneme can be conceived of as representing a family of related phones that speakers of a language think of as being categorically the same. While the notion of phonemes has been controversial among linguists (see [40.34] for some history), it has proven a useful abstraction for speech processing. *Ladefoged* [40.35] speculates that 'there are probably about 600 different consonants' (p. 194) across the languages of the world; vowels and suprasegmental distinctions (such as tone) are not quite as numerous.

Phonetic symbols provide a way to transcribe the sounds of spoken languages. There are several phonetic alphabets; one commonly used by linguists is the international phonetic alphabet (IPA). This alphabet, including a set of diacritic modifiers, was established as a standard by the International Phonetic Association (IPA) in order to provide an accurate representational system for transcribing the speech sounds of all languages (the IPA website is at [40.36]). The goal for the IPA is to provide a representation for all of the phonemes expressed in all human languages, such that separate symbols are used for two sounds only if there exists a language for which these two sounds are distinguished phonemically. For the most recent update of the IPA, see [40.37].

Given the availability of a phonetic or phonemic representation for language sounds, there are various ways in which this information can help to differentiate among languages, including:

1. *Phonemic inventory:* It is possible to distinguish some languages from one another based on the presence of a phoneme or phone that appears in one language but not the other. Phoneme inventories range from a low of 11 (for Rotokas, a Papuan language; see [40.38]) to a high of a hundred or more (in certain Khoisan languages of southern Africa [40.39]). It is likely that no two languages share exactly the same phoneme inventory. Furthermore, even if two languages were to share a common set of phonemes, the phonemes themselves will likely differ in their relative frequency patterns [40.22].
2. *Broad class inventory:* Patterns of broad phonetic categories (e.g., vowels, fricatives, plosives, nasals, and liquids) have also been utilized to distinguish among languages in an attempt to avoid the need for fine phonetic recognition. For example, *Muthusamy* [40.22] evaluated the use of seven broad phonetic categories (vowel, fricative, stop, closure or silence, prevocalic sonorant, intervocalic sonorant, and postvocalic sonorant). However, *Hazen* [40.23] found that, even though the broad class phone

recognizers tend to be more accurate than the more-traditional finer-grained phone recognizers, using broad class inventories leads to a lower-accuracy language identification system.

3. *Phonotactics:* Phonotactics refers to the arrangements of phones or phonemes within words. Even if two languages were to share a common phoneme inventory, it is likely that they would differ in phonotactics. The first proponents of using phonotactic features were *House* and *Neuburg* [40.40], who believed accurate language identification could be achieved by making use of the statistics of the linguistic events in an utterance, in particular, language-specific phonetic sequence constraints. (They suggested using a sequence of broad phonetic classes as a way of obtaining more-reliable feature extraction across languages, although this has been found to result in less accurate language identification systems [40.23].) An implementation using this approach would typically involve several steps, where the first is to map an utterance to a sequence of phonetic labels (i. e., tokenization into phones), which would then be used to identify the language based on the observed n-grams.
   There are several variants of this approach to audio based language identification. Phone-based LID uses a single-language phone recognizer trained for some arbitrary language with sufficient resources (not necessarily one of the target languages) to tokenize the speech input into *phones* for that language, followed by the use of n-gram probabilistic language models (one for each target language) to calculate the likelihood that the symbol sequence was produced in each of the target languages, with the highest likelihood language being selected. The parallel phone recognition followed by language model (PPRLM) is similar except that it uses phone recognizers from several languages, together with some method to normalize and combine the results from the parallel streams [40.41].
   A third variant would be to train a phone recognizer on a broad-coverage phonetic database (such as that in [40.35]). To reduce the time and cost to develop speech systems in a new language, researchers have been investigating the development and use of a multilingual phone set that represents sounds across the languages to be modeled. *Schultz* and *Waibel* [40.42] in their research on multilingual speech recognition defined a phone set covering 12 languages. They assume that *the articulatory representations of phonemes are so similar across languages, that phonemes can be considered as units which are independent from the underlying language* (p. 1) [40.43].
   Hazen and *Zue* [40.23, 32] developed a LID system that was based on 87 language-independent phone units that were obtained by hand clustering approximately 900 phone labels found in training transcriptions. Researchers at the computer sciences laboratory for mechanics and engineering sciences (LIMSI) have also been investigating the use of a *universal* phone set [40.44, 45] and have attempted to identify objective acoustic criteria for clustering the language-dependent phones. *Corredor-Ardoy* et al. [40.46], found that their LID system using a language-independent set (based on clustering) performed as well as their best methods using language-dependent phones. *Ma* and *Li* [40.47] evaluated the use of a universal sound recognizer to transcribe utterances into a sequence of sound symbols that act as a common phone set for all of the languages to be identified. They then used statistics related to the large-span co-occurrence of the sound patterns, which they dubbed the bag-of-sounds approach, to identify the language of the utterance.
4. *Articulatory features:* Speech can be characterized by parallel streams of articulatory features which are used in concert to produce a sequence of phonemes. These features could be exploited to differentiate one language from another. For example, the phoneme /t/ can be realized either with or without aspiration, with a dental or alveolar closure, and with lips rounded or not [40.23]. *Kirchhoff* and *Parandekar* [40.48] utilized a set of pseudoarticulatory classes that were designed to capture characteristics of the speech production process, including: manner of articulation, consonantal place of articulation, vocalic place of articulation, lip rounding, front–back tongue position, voicing, and nasality. They developed an alternative approach to audio-based language identification that was based on the use of parallel streams of these subphonemic events together with modeling of some of the statistical dependencies between the streams.

Syllable Structure. A syllable is a unit of pronunciation that is larger than a single sound, composed of a peak of sonority (usually a vowel, but sometimes a sonorous consonant), bordered by troughs of sonority (typically consonants) [40.33]. Languages can be characterized by common syllable types, typically defined in terms of sequences of consonants (C) and vowels (V). However, it should be noted that breaking sequences of Cs and

Vs into syllables – the process of syllabification – is often difficult, and even controversial [40.49, 50]. However, since languages generally allow or disallow certain types of syllable structures (e.g., Slavic languages often have complex consonant clusters in contrast to Asian languages), this representation may help in discriminating among languages. For example CCCCVC is a valid syllable type in Russian, but not in most other languages.

Zhu et al. [40.51] developed a LID system whose acoustic decoder produces syllable streams, which they then used in syllable-based (rather than phone-based) *n*-gram language models. Accents of foreign speakers of English manifest themselves differently given their position within the syllable, a fact that has been used to improve accent identification [40.52].

Prosodic Information. The duration, pitch, and stress patterns in one language often differ from another. For example, different languages have distinct intonation patterns. In stress languages, pitch is often one correlate of stress used to mark syllable prominence in words (and in pitch accent languages, such as Swedish, the primary correlate); whereas, in tone languages, a change in the meaning of a word is signalled by the tone on the syllables or other tone-bearing units (e.g., Mandarin Chinese or Thai). For stress languages, patterns of stress can provide an important cue for discriminating between two languages. Some of the stress patterns are initial stress (e.g., Hungarian), penultimate stress (e.g., Polish or Spanish), final stress (e.g., French or Turkish), and mixed stress (e.g., Russian or Greek). Prosodic cues of duration can also be potentially useful. For example, some languages (such as Finnish) distinguish long and short vowels and/or consonants.

One ostensibly useful typology investigated by both linguists and psychologists involves the rhythm of a language, where a distinction is made between stress-timed languages (in which stressed syllables are longer than unstressed syllables, all else being equal, e.g., English), syllable-timed (each syllable has comparable time duration, e.g., French), and mora-timed (each mora has essentially constant duration, e.g., Japanese) languages [40.53]. Although this classification remains controversial [40.54], rhythmic modeling has been investigated by *Rouas* et al. [40.55, 56] for language identification. Their rhythm model was able to discriminate fairly accurately between languages on a read speech corpus, but less well on a spontaneous speech corpus [40.55]. Indeed, extracting reliable prosodic cues from spontaneous speech is a challenge due to its variability.

In language identification systems that utilize prosodic features, these features are typically combined with other knowledge sources to achieve reasonable accuracy. *Muthusamy* [40.22] was able to incorporate pitch variation, duration, and syllable rate features into his LID model. *Hazen* and *Zue* [40.23, 32] integrated duration and pitch information into their LID model, with the duration model being more accurate on its own than the pitch model. *Tong* et al. [40.57] successfully integrated prosodic features (i. e., duration and pitch) with spectrum, phonotactic, and bag-of-sounds features, where pitch variation and phoneme duration were especially useful for short speech segments.

Lexical Information. Each language has its own vocabulary, which should help in identifying a language more reliably. For speech inputs, this would require the availability of a speech recognition system for each of the candidate languages, along with the requisite training, tuning, and evaluation materials needed to ensure the speech model is adequate. *Schultz* et al. [40.58, 59] developed a LID system for four languages based on large-vocabulary continuous speech recognition (LVCSR). They found that word-based systems with trigram word language modeling significantly outperform phone-based systems with trigram phone modeling on the four-language task, suggesting that the lexical level provides language discrimination ability, even when word error rates are fairly high. *Matrouf* et al. [40.60] found that incorporating lexical information with a phone-based approach yielded relative error reductions of 15–30%, and that increasing lexical coverage for a language had a positive effect on system performance. *Hieronymus* and *Kadambe* [40.61] constructed a LID system based on LVCSR for five languages (English, German, Japanese, Mandarin Chinese, and Spanish), obtaining 81% and 88% correct identification given 10 and 50 second utterances, respectively, without using confidence measures and 93% and 98% correct with confidence measures. Although each language clearly has its own vocabulary that enables language identification systems to discriminate among a candidate set of languages more effectively, for spoken languages, this information is generally quite expensive to obtain and use.

Morphology. Morphology is a branch of grammar that investigates the structure of words [40.33]. The field of morphology is divided into two subfields: inflectional morphology, which investigates affixes that signal grammatical relationships that do not change the grammatical

class of a word (e.g., affixes marking tense, number, and case) and derivational morphology, which focuses on word formation involving affixes, such as *-ment*, that can be used to create a new word form with a possibly different grammar class, such as the noun *amendment* derived from the verb *amend*. Since languages form words in a variety of different ways, morphology could provide an excellent cue for automatic language identification. For example, one could use common suffixes to discriminate among some of the Romance languages (e.g., *-ment* in French, *-miento* in Spanish, *-mento* in Portugese, and *-mente* in Italian). Although in speech the morphology of a word is covered in part by phonotactics, with morphological knowledge of its candidate languages, a LID system could focus on specific portions of words when discriminating between two languages.

Syntax. Languages and dialects also differ in the ways that words are arranged to create a sentence. They differ in the presence or absence of words with different parts of speech, as well as in the ways that words are marked for various types of roles in a sentence. In conjunction with other kinds of information (e.g., accents), errors in grammatical usage could provide a helpful cue for identifying the first language of someone speaking a second language. For example, someone who learned Mandarin as a first language would tend to make determiner (deletion, substitution, and insertion) and agreement errors in English or German.

Languages also often differ from each other in the word order of a sentence's subject (S), verb (V), and object (O). For example, English is considered to be an SVO language because the subject typically appears before the verb, which occurs before the object; whereas, Japanese is an SOV language. Even if two languages have the same word form, it is likely that the word would appear in very different word contexts across languages. Although words may be sufficient to distinguish among languages, syntax could play an especially important role for discriminating among dialects of a language.

Language and dialect identification systems that are based on LVCSR would utilize an acoustic model, a dictionary, and a language model for each language or dialect in the candidate set. The language models utilize word co-occurrence statistics that capture some aspects of the candidate language's syntactic structure. These systems could be expanded to utilize syntax more directly by using structured language models (e.g., [40.62, 63]).

Other Information. Higher-level knowledge sources such as semantics and pragmatics are rarely used by audio-based LID systems, although this type of knowledge could potentially help. In addition, information about the source of the speech data (e.g., country of origin of a broadcast news show) could be used to narrow down the language choices.

Research on automatic language identification suggests that the more knowledge that goes into a decision about which language corresponds to an audio sample, the greater the accuracy; hence, knowledge integration is important. However, there is a trade-off between accuracy and efficiency, and furthermore, there is a need for resources to support the knowledge brought into the automatic system. Much of the work has struck an engineering balance in addressing this problem; they use the resources that can be obtained simply and reliably for the set of languages to be discriminated among.

It is important to note that the length of an audio sample (the amount of speech available) will impact the knowledge sources that can be reliably used in determining its language. The smaller the samples used, the less likely that a key piece of higher-level knowledge will be available to discriminate a particular language from the others.

40.4 Human Language Identification

A human who knows the language being spoken is capable of positively identifying short samples of speech quickly and accurately. Even if they do not know the language, people can make sound decisions about the identity of a language given some exposure to the language, and with some training about cues that differentiate a set of candidate languages, this capability can be improved and expanded.

There have been several experiments reported in the literature that consider human ability in a scenario where the decision is based on a combination of their prior knowledge about certain languages (a variable that is difficult to control) and a limited amount of online training for the languages being identified. *Muthusamy* et al. [40.22, 64] had human subjects listen to short samples of 10 different languages and guess the language

of the sample. In general, he found that familiarity with a language was an important factor affecting accuracy, as was the length of the speech sample (with longer samples leading to greater accuracy). Subjects were able to improve their LID accuracy only slightly over time with feedback, but they were quite aware of the cues they used for language discrimination (e.g., phonemic inventory, word spotting, and prosody).

Maddieson and *Vasilescu* [40.65] examined the effect of exposure to academic linguistics on language identification accuracy of five language. Subjects with more than *passing* familiarity with the language were excluded from the study. They found that prior casual exposure to a language and linguistic education level (ranging from no linguistic training to a PhD) were not effective predictors of performance on a five-language identification task; however, they found that linguistic training did predict improved performance on a language discrimination task (in which subjects were asked to decide if a sample was one of the five target languages, similar to one of those languages, or unlike them).

Several human studies have been conducted in an attempt to determine what types of information people can effectively utilize when making decisions about the identity of a language. These experiments involve the presentation of speech stimuli that were obtained by modifying the speech samples presented to the subjects. For example, *Mori* et al. [40.66] found that their subjects were able to identify two languages (Japanese and English) fairly reliably even when segmental information was reduced using signal editing techniques. They argue that other cues such as intensity and pitch are being used to make these judgments. *Navratil* [40.67] evaluated the importance of various types of knowledge, including lexical, phonotactic, and prosodic, by humans asked to identify the language (Chinese, English, French, German, or Japanese) of a speech sample. Subjects were presented unaltered speech samples, samples containing randomly ordered syllables from speech samples, and samples for which the spectral shape was flattened and vocal-tract information removed (leaving F_0 and amplitude). Navratil found that humans on six second samples were far more accurate at identifying unaltered speech samples (96%) than samples with shuffled syllables (73.9%), and were more accurate with the shuffled syllable samples than samples with only prosodic cues remaining (49.4%). Based on these experiments, it appears that the lexical and phonotactic information provides discriminative information that is used more reliably by humans.

None of the subjects in these listening experiments were explicitly trained to identify cues to discriminate one language from another, hence, these experiments did not explore the full range of human capability that could be achieved with training on a particular set of languages to be identified. People can identify the language of an audio input fairly reliably when they do not speak/understand the language by being taught to use a variety of cues that are discriminative for a language or language family. Some combination of the following sorts of clues can be used to identify a particular language or to narrow the possibilities down to a smaller set of languages:

- general impression ('gestalt'), i. e., what a given language sounds like, which may help narrow the language down to a geographic area or a language family
- stress patterns, where these may be most reliably discerned at pause boundaries or on assimilated loan words
- vowel and/or consonant durations
- the presence or absence of nasalization on (some) vowels
- the presence or absence of lexical tone
- syllable structure, particularly the presence or absence of consonant clusters
- the presence of unusual sounds, such as front rounded vowels, glottalized consonants, clicks, or retroflexed consonants
- reduplication (particularly full-word reduplication)
- the presence of common words, particularly short high-frequency words that are easily recognizable (e.g., determiners, prepositions)

Some of these features would obviously be difficult for computers to use, including the 'gestalt' of the language or detecting reduplication. Other features are similar to what programs doing spoken language identification already utilize, e.g., the use of consonant clusters (phonotactics). Still others of these features suggest possible directions for future work in automated language ID, for example recognizing unusual sounds.

There is also a more-general difference in the methodology used by humans and speech-based LID algorithms. Most speech-based LID systems do not utilize the high-level knowledge that people tend to use. Also, most of the automatic LID algorithms tend to process and combine evidence from the entire speech stream when making a decision about the identity of a language; whereas, humans rely on very specific cues taken from small portions of the sample to refine their hypotheses.

For example, two of the cues used by human experts for LID on texts are the pairings of diacritics with base characters and the presence of short (but common) words. Diacritics are especially difficult for image-based language identification systems, which can have difficulty differentiating them from noise in the image. In contrast, the primary cue used by computers for computer encoded text or in speech is the statistical frequency character or phone *n*-grams. *n*-grams would be hard for people to learn, with the exception of unusual single letters or sounds and morphemes which are cognate with English morphemes; anything more probably requires the user to refer to a *cheat sheet*. Even harder for people is computing the statistics of *n*-grams; judgments with a finer granularity than *frequent* or *rare* would be difficult for people to make.

In summary, while it is possible for computers to make use of more of the characteristics of languages than humans use to identify them, it is probably not practical to teach people to use the methodology used by automatic LID systems to do language identification.

40.5 Text as a Source of Information on Spoken Languages

Given the difficulty in building comparable speech corpora for a significant number of languages, and in particular for rare languages, another source of information that might be mined to learn more about the spoken form of a language is its written form. (While most unwritten languages have small speaker populations, and the total number of speakers of unwritten languages is much smaller than the number of speakers of written languages, there are still significant numbers of languages of the world – perhaps more than half – which are unwritten.) It is comparatively easy to build a text corpus for a written language; however, there are several issues that affect whether the textual data will be useful in characterizing the spoken language. An important issue is how close the written language is to its typical spoken form. There are many aspects to this question, but two of the most significant issues are diglossia and complex orthographies.

A diglossic language situation exists when two (or more) forms of a language coexist, and the forms diverge to a significant degree; typically one form is perceived as *high* (correct), and the other as *low* (vulgar). For our purposes, diglossia is relevant when the high and low varieties correspond to the written and the spoken languages respectively, where they differ significantly in style and vocabulary. Tamil is a typical example; the written form of the language is considered classical, and the spoken forms (there is more than one dialect) are considered low. In such a situation, written corpora may not be representative of the spoken form of the language.

Arabic is another important example of a diglossic situation: the written form, known as modern standard Arabic (MSA), is taught in schools; whereas, the spoken varieties – of which the Ethnologue [40.1] lists nearly 40 – are strikingly different from MSA in vocabulary, morphology, and phonology (the latter with reference to how MSA is generally read out loud, for instance on news broadcasts).

As for the complex orthography issue, for our purposes a complex orthography is one in which the written forms of words do not have a direct mapping to the spoken forms. English is a notorious example of this, and so to a lesser extent is French. A related issue is orthographies which undermark phonemic distinctions in the spoken language. Written Arabic is an example, since the short vowels are not normally written, resulting in significant ambiguity: multiple morphological analyses, each corresponding to different pronunciations, are possible for a large percentage of the words in running Arabic (MSA) text.

Many orthographies that are complex today are so only because the spoken language has changed faster than the written language. Orthographies that have been developed in the recent past tend to be less complex, i. e., they usually map more or less directly to the phonemes of the spoken language (or to a standard dialect of the language), and can therefore be said to be *phonemic*. However, some otherwise phonemic modern orthographies fail to discriminate a subset of the phonemic contrasts of the language, whether in practice or in principle. For example, while Yoruba is supposed to be written with tone marks for the high and low tones (and optionally for the mid tone), in practice these are often omitted, as are the dots under the Yoruba letters 'e' and 'o', intended to indicate a more-open vowel than the same letters without the dot. Finally, some orthographies omit certain phonemic distinctions (e.g., tone) on the principle that the distinction in question is not important enough in that language.

Assuming that a language's orthography is close to phonemic, it is also necessary to know the mapping from individual characters (or sequences of characters) to a phonemic representation. This is because orthographies do not always use the letters in a standard way. In Hungarian, for example, the letter 's' represents an alveopalatal (like the digraph 'sh' in English), and the digraph 'sz' represents an alveolar fricative (like the English letter 's' in most words); Polish represents these sounds in exactly the opposite way. Fortunately, this information about alphabets can generally be obtained.

In summary, then, it would be possible at least in principle to expand the number of spoken languages that can be investigated and characterized linguistically by using more readily obtainable text corpora in place of speech corpora. Whether these corpora can be used to enhance speech-based LID systems is an open question.

40.6 Summary

In this chapter, we have discussed a variety of knowledge sources that could be utilized to characterize a spoken language in order to distinguish it automatically from other languages. We discussed differences between a language and a dialect, and also described some of the variability in a language that could potentially challenge LID algorithms. Currently available corpora for evaluating language identification systems have only *scratched the surface* of the possible space of languages that could be investigated. Most state-of-the-art systems are able to detect tens of languages as opposed to the 100 or more that would be required by Language Line Services. It remains to be seen whether acoustic- and phonotactic-based systems will effectively scale up to handle these one hundred plus languages. As the number of languages and dialects increase, it is likely that systems will need to utilize more linguistic insight to achieve accuracies comparable to those obtained over a smaller set of languages. These larger systems could utilize more lexical information to achieve target accuracies; however, this knowledge source comes with a high cost for system development.

We also discussed several experiments reported in the literature that investigated a person's ability to accurately identify a spoken language. None of these studies involved a situation where participants were trained to accurately identify a language based on salient language-specific high-frequency cues. We enumerated some cues that have been commonly used, some of which overlap with features used by automatic LID systems. However, since some of the cues that humans use would be difficult to incorporate into an automatic LID system (e.g., a general impression of the language), it is interesting to contemplate whether there would be some way to build a partnership between trained humans and LID systems.

We ended this chapter by discussing conditions under which textual materials could potentially augment our knowledge of a spoken language, in particular, a rare language. There are a number of factors that impact the correspondence between spoken and written language forms; however, if there is a good correspondence, the written form could be used to gain a deeper understanding of the kinds of features that would help discriminate the language from others.

References

40.1 R.G. Gordon Jr. (ed): *Ethnologue: Languages of the World*, 15th edn. (SIL International, Dallas 2005), online version: http://www.ethnologue.com

40.2 Language Line Services, http://www.languageline.com

40.3 A.G. Chejne: *The Arabic Language Its Role in History* (University of Minnesota Press, Minneapolis 1969)

40.4 N. Haeri: *Sacred Language Ordinary People: Dilemmas of Culture and Politics in Egypt* (Palgrave MacMillan, New York 2003)

40.5 C.P. Masica: *The Indo-Aryan Languages* (Cambridge Univ. Press, Cambridge 1991)

40.6 B.A. Fennel: *A History Of English: a Sociolinguistic Approach*, Blackwell Textbooks in linguistics, Vol. 17 (Blackwell, Malden 2001)

40.7 P. Constable, G. Simons: An Analysis of ISO 639: Preparing the way for advancements in language identification standards, 20th International Unicode Conference (2002), available as http://www.ethnologue.com/14/iso639/An_analysis_of_ISO_639.pdf

40.8 ISO/DIS 639-3, http://www.sil.org/iso639-3/default.asp

40.9 D. Barton: *Literacy: An Introduction to the Ecology of Language* (Blackwell, Malden 1994)

40.10 C.W. Kreidler: *Describing Spoken English: An Introduction* (Routledge, London 1997)

40.11 A. McMahon: *Lexical Phonology and the History of English*, Cambridge Studies in Linguistics 91 (Cambridge Univ. Press, Cambridge 2000)

40.12 The SCOTS Project, http://www.scottishcorpus.ac.uk

40.13 R. Sproat, T.F. Zheng, L. Gu, D. Jurafsky, I. Shanfran, J. Li, Y. Zheng, H. Zhou, Y. Su, S. Tsakalidis, P. Bramsen, D. Kirsch: *Dialectical Chinese Speech Recognition: Final Technical Report* (Johns Hopkins University, Baltimore 2004), CLSP

40.14 J.C. Wells: *Accents of English*, Vol. 3 (Cambridge Univ. Press, Cambridge 1982)

40.15 P. Trudgill, J. Hannah: *International English: A Guide to Varieties of Standard English* (Oxford Univ. Press, New York 2002)

40.16 A. Sakaguchi: Towards a clarification of the function and status of international planned languages. In: *Status and Function of Language and Language Varieties*, ed. by U. Ammon (Walter de Gruyter, Berlin 1989) p. 399

40.17 W. Byrne, M. Finke, S. Khudanpur, J. McDonough, H.J. Nock, M. Riley, M. Saralar, C. Wooters, G. Zavaliagkos: Pronunciation modelling using a handlabelled corpus for conversational speech recognition, Proc. ICASSP (1998) pp. 313–316

40.18 E. Fosler-Lussier, N. Morgan: Effects of speaking rate and word frequency on conversational pronunciations, Speech Commun. **29**, 137–158 (1999)

40.19 S. Greenberg: Speaking in shorthand – A syllablecentric perspective for understanding pronunciation variation, Speech Commun. **29**, 159–176 (1999)

40.20 Multi-Language Telephone Speech Corpus Distribution, http://www.ldc.upenn.edu/Catalog/readme_files/ogi readme.html, January 1994.

40.21 OGI Multilanguage Corpus, http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC94S17

40.22 Y.K. Muthusamy: *A Segmental Approach to Automatic Language Identification* (Oregon Graduate Institute of Science and Technology, Beaverton 1993), Ph.D. Thesis

40.23 T.J. Hazen: *Automatic Language Identification Using a Segment-Based Approach* (MIT, Cambridge 1993), Masters Thesis

40.24 LDC Projects, http://www.ldc.upenn.edu/Catalog/project_index.jsp

40.25 CLSU: 22 Languages Corpus, http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2005S26

40.26 MIXER Telephone Study, http://mixer.ldc.upenn.edu

40.27 D. Cimarusti, R.B. Ives: Development of an automatic identification system of spoken languages: Phase 1, Proc. ICASSP (1982) pp. 1661–1663

40.28 P.A. Torres-Carrasquillo, D.A. Reynolds, J.R. Deller Jr.: Language identification using Gaussian mixture model tokenization, Proc. ICASSP (2002) pp. 757–760

40.29 M.A. Zissman: Automatic Language Identification Using Gaussian Mixture and Hidden Markov Models, Proc. ICASSP (1993) pp. 399–402

40.30 K.K. Wong, M. Siu: Automatic language identification using discrete hidden Markov model, Interspeech (2004) pp. 1633–1636

40.31 W.M. Campbell, E. Singer, P.A. Torres-Carrasquillo, D.A. Reynolds: Language recognition with support vector machines, Proceedings of Odyssey: The Speaker and Language Recognition Workshop (2004) pp. 41–44

40.32 T.J. Hazen, V. Zue: Segment-based automatic language identification, J. Acoust. Soc. Am. **101**(4), 2323–2331 (1997)

40.33 D. Crystal: *A Dictionary of Linguistics and Phonetics*, 2nd edn. (Basil Blackwell, Oxford: 1985)

40.34 M. Kenstowicz: Generative Phonology. In: *Encyclopedia of Language and Linguistics*, ed. by K. Brown (Elsevier, Amsterdam 2005), 2nd edn.

40.35 P. Ladefoged: *Vowels and Consonants: An Introduction to the Sounds of Languages* (Blackwell, Oxford 2005)

40.36 The International Phonetic Association, http://www.arts.gla.ac.uk/IPA/index.html

40.37 Reproduction of the International Phonetic Alphabet, http://www.arts.gla.ac.uk/IPA/ipachart.html , 2005.

40.38 I. Maddieson: *Sound Patterns of Language* (Cambridge Univ. Press, Cambridge 1984)

40.39 A. Traill: *Phonetic and Phonological Studies of !Xóõ bushman* (Helmut Buske, Hamburg 1985)

40.40 A.S. House, E.P. Neuberg: Toward automatic identification of the languages of an utterance: preliminary methodological considerations, J. Acoust. Soc. Am. **62**(3), 708–713 (1977)

40.41 M.A. Zissman: Comparison of four approaches to automatic language identification of telephone speech, IEEE Trans. Speech Audio Process. **4**(1), 31–44 (1996)

40.42 T. Schultz, A. Waibel: Language independent and language adaptive acoustic modeling for speech recognition, Speech Commun. **35**(1-2), 31–51 (2001)

40.43 A. Black, T. Schultz: Speaker clustering for multilingual synthesis, Proceedings of the ISCA Tutorial and Research Workshop on Multilingual Speech and Language Processing (2006)

40.44 M. Adda-Decker, F. Antoine, P.B. de Mareuil, I. Vasilescu, L. Lamel, J. Vaissiere, E. Geoffrois, J.-S. Liénard: Phonetic knowledge, phonotactics and perceptual validation for automatic language identification, International Congress of Phonetic Sciences (2003)

40.45 P.B. de Mareüil, C. Corredor-Ardoy, M. Adda-Decker: Multi-lingual automatic phoneme clustering, Int. Congress Phonetic Sci. (1999) pp. 1209–1213

40.46 C. Corredor-Ardoy, J.L. Gauvain, M. Adda-Decker, L. Lamel: Language Identification with Language-independent Acoustic Models, Proc. Eurospeech (1997) pp. 355–358

40.47 B. Ma, H. Li: Spoken language identification using bag-of-sounds, International Conference on Chinese Computing (2005)

40.48 K. Kirchhoff, S. Parandekar: Multi-stream statistical language modeling with application to automatic language identification, Proceedings of the 7th European Conference on Speech Communication and Technology Proceedings of Eurospeech (2001) pp. 803–806

40.49 J. Blevins: The syllable in phonological theory. In: *The Handbook of Phonological Theory*, Blackwell Handbooks in Linguistics, Vol. 1, ed. by J.A. Goldsmith (Blackwell, Oxford 1995) pp. 206–244

40.50 M. Kenstowicz: *Phonology in Generative Grammar*, Blackwell Textbooks in Linguistics, Vol. 7 (Blackwell, Oxford 1994)

40.51 D. Zhu, M. Adda-Decker, F. Antoine: Different size multilingual phone inventories and context-dependent acoustic models for language identification, Interspeech (2005) pp. 2833–2836

40.52 K. Berkling, M. Zissman, J. Vonwiller, C. Cleirigh: Improving accent identification through knowledge of English syllable structure, Proceedings of the 5th International Conference on Spoken Language Processing (1998) pp. 89–92

40.53 E. Grabe, E.L. Low: Durational variability in speech and the rhythm class hypothesis. In: *Laboratory Phonology*, ed. by C. Gussenhoven, N. Warner (Mouton de Gruyter, Berlin 2002) pp. 515–546

40.54 R.M. Dauer: Stress-timing and syllable-timing reanalysed, J. Phonet. **11**, 51–62 (1983)

40.55 J. Rouas, J. Farinas, F. Pellegrino, R. Andre-Obrecht: Modeling prosody for language identification on read and spontaneous speech, Proc. ICASSP (2003) pp. 40–43, vol. 6

40.56 J. Rouas, J. Farinas, F. Pellegrino, R. André-Obrecht: Rhythmic unit extraction and modelling for automatic language identification, Speech Commun. **47**(4), 436–456 (2005)

40.57 R. Tong, B. Ma, D. Zhu, H. Li, E.S. Chang: Integrating acoustic, prosodic and phonotactic features for spoken language identification, Proc. ICASSP (2006) pp. 205–208

40.58 T. Schultz, I. Rogina, A. Waibel: Experiments with LVCSR based language identification, Proceedings of the Speech Symposium SRS XV (1995)

40.59 T. Schultz, I. Rogina, A. Waibel: LVCSR-based language identification, Proc. ICASSP (1996) pp. 781–784

40.60 D. Matrouf, M. Adda-Decker, L. Lamel, J. Gauvain: Language identification incorporating lexical information, Proceedings of the 5th International Conference on Spoken Language Processing (1998) pp. 181–184

40.61 J. Hieronymus, S. Kadambe: Robust spoken language identification using large vocabulary speech recognition, Proc. ICASSP (1997) pp. 779–782

40.62 C. Chelba: *Exploiting Syntactic Structure for Natural Language Modeling* (Johns Hopkins University, Baltimore 2000), Ph.D. thesis

40.63 W. Wang, M. Harper: The SuperARV language model: Investigating the effectiveness of tightly integrating multiple knowledge sources, Proceedings of Conference of Empirical Methods in Natural Language Processing (2002) pp. 238–247

40.64 Y.K. Muthusamy, E. Barnard, R.A. Cole: Reviewing automatic language identification, IEEE Signal Process. Mag. **11**(4), 33–41 (1994)

40.65 I. Maddieson, I. Vasilescu: Factors in human language identification, Proceedings of the 5th International Conference on Spoken Language Processing (2002) pp. 85–88

40.66 K. Mori, N. Toba, T. Harada, T. Arai, M. Komatsu, M. Aoyagi, Y. Murahara: Human language identification with reduced spectral information, Proceedings of the 6th European Conference on Speech Communication and Technology (1999) pp. 391–394

40.67 J. Navratil: Spoken language recognition – A step towards multilinguality in speech processing, IEEE Trans. Speech Audio Process. **9**(6), 678–685 (2001)

42. Vector-Based Spoken Language Classification

H. Li, B. Ma, C.-H. Lee

This chapter presents a vector space characterization (VSC) approach to automatic spoken language classification. It is assumed that the space of all spoken utterances can be represented by a universal set of fundamental acoustic units common to all languages. We address research issues related to defining the set of fundamental acoustic units, modeling these units, transcribing speech utterances with these unit models and designing vector-based decision rules for spoken language classification. The proposed VSC approach is evaluated on the 1996 and 2003 National Institute of Standards and Technology (NIST) language recognition evaluation tasks. It is shown that the VSC framework is capable of incorporating any combination of existing vector-based feature representations and classifier designs. We will demonstrate that the VSC-based classification systems achieve competitively low error rates for both spoken language identification and verification.

The chapter is organized as follows. In Sect. 42.1, we introduce the concept of vector space characterization of spoken utterance and establish the notion of acoustic letter, acoustic word and spoken document. In Sect. 42.2 we discuss acoustic segment modeling in relation to augmented phoneme inventory. In Sect. 42.3, we discuss voice tokenization and spoken document vectorization. In Sect. 42.4, we discuss vector-based classifier design strategies. In Sect. 42.5, we report several experiments as the case study of classifier design, and the analytic study of front- and back-end. Finally in Sect. 42.6, we summarize the discussions.

Spoken language classification is the process of determining the language identity of a given spoken utterance. Like many frontiers in pattern recognition, spoken language classification has been formulated in the framework of statistical modeling, where a model is built for each spoken language by fitting the model parameters to maximize the observed data likelihood. A spoken language classification system typically consists of a front- and a back-end. The front-end extracts features in order to characterize spoken languages, while the back-end makes classification decision based on the feature pattern of a test utterance.

When human beings are constantly exposed to a language without being given any linguistic knowledge, they learn to determine the language identity by perceiving some of the language cues. This motivates us to explore useful language cues from phonotactic statistics of spoken languages. Suppose that the world's languages can be characterized by a universal set of acoustic units. Any speech segment can now be considered as a realization of a concatenation of such units analogous to representing a text passage as a sequence of basic symbols, such as English letters or Chinese characters. Using a collection

of models, one for each unit, a given speech utterance can be automatically transcribed into a sequence of sound alphabets or *acoustic letters* By grouping adjacent *acoustic letters* to form *acoustic words* we can convert any collection of speech segments into a *spoken document*. Similar to representing a text document as a term vector – as is done in information retrieval and text categorization – a spoken document can now be represented by a vector with key terms defined as the set of all acoustic words. This process is also referred to as vectorization, which leads us into several interesting topics, such as selection of acoustic letters and acoustic words, and dimensionality reduction of term vector.

With recent advances in machine learning techniques, vector space modeling has emerged as a promising alternative solution to multiclass speech classification problem. The vector space approach inherits several attractive properties that we discover in text-based information retrieval. For example, it handles the fusion of different types of language cues in a high-dimensional vector seamlessly. As opposed to similarity-based likelihood measurement, the vector space approach is motivated by discriminative training, which is aimed at minimizing misclassification error. In this chapter, we are interested in practical issues related to vectorization of spoken document and vector-based classifier design.

42.1 Vector Space Characterization

In a typical setting, spoken language classification is usually formulated as a probabilistic pattern recognition problem [42.1] in which the a posterior probability, $P(X|\lambda_l)$, of the l-th language to be considered with a model λ_l given an unknown utterance X, is computed. To make a decision, the language that gives the maximum $P(X|\lambda_l)$ is usually identified as the target language. Many algorithms developed in automatic speech and speaker recognition [42.2] have been adopted and extended to language recognition. Recent advances in acoustic and language modeling [42.3] have also contributed to the technological progress of pattern recognition approaches to spoken language classification.

On another front vector space characterization has become a prevailing paradigm in the information retrieval (IR) community since its introduction in the early 1970s [42.4]. Inspired by recent advances in machine learning, a wide variety of VSC approaches have been adopted successfully to text categorization (TC) [42.5], which essentially refers to the assigning of a category or a topic to a given text document based on the frequencies of co-occurrences of certain key terms [42.6] in the particular category or topic of interest. A similar VSC treatment of spoken documents has also been attempted [42.7]. Since the feature vectors to be considered here are often very high dimensional, sometimes in tens or even hundreds of thousands in some cases, a probabilistic characterization of these vectors is usually not an easy task due to several factors: the lack of training data, the difficulty to specify a good model, and the curse of dimensionality in the estimation of so many parameters with so little data. Nonetheless, vector-based classifier design has some of its own attractive properties that does deserve some attention even without using conventional statistical modeling techniques.

In this chapter, we explore VSC characterization of spoken utterances and the VSC-based spoken language classification system design. We consider that a particular spoken language will always contain a set of high-frequency function words, prefixes, and suffixes, which are realized as acoustic substrings in spoken utterances. Individually, those substrings may be shared across languages. Collectively, the pattern of their co-occurrences can facilitate the discrimination of one language from another.

Suppose that an utterance X, represented by a sequence of speech feature vectors $\boldsymbol{O}$, is decoded or tokenized, into a *spoken document*, $T(X)$, consisting of a series of I acoustic units, $T(X) = \{t_1 \ldots, t_i \ldots, t_I\}$, each unit is drawn from a universal inventory, $U = \{u_1, \ldots, u_j, \ldots u_J\}$, of J acoustic letters shared by all the spoken languages to be considered, such that $t_i \in U$. In the following, we will refer to the process of decoding speech into spoken documents as *tokenization*. We are then able to establish a collection of acoustic words by grouping units occurring in consecutive orders to obtain a vocabulary of M distinct words, $W = \{w_1, \ldots, w_m, \ldots w_M\}$, such that each w_m can be a single-letter word like (u_j), a double-letter word like $(u_j u_k)$, a triple-letter word like $(u_j u_k u_l)$, and so on. Usually the vocabulary size, M, is equal to the total number of n-gram patterns needed to form words, e.g., $M = J + J \times J + J \times J \times J$ if only up to three tokens are considered as a valid acoustic word. Next, we can use

some form of function $f(w_m)$, such as latent semantic indexing (LSI) [42.8], to evaluate the significance of having the word w_m in the document, $T(X)$. We are now ready to establish an M-dimension feature vector, $\boldsymbol{v} = [f(w_1), \ldots, f(w_m), \ldots f(w_M)]^{\mathrm{T}}$ with $\boldsymbol{x}^{\mathrm{T}}$ denoting the transpose of vector $\boldsymbol{x}$, for each spoken document.

It is clear that we need a not-too-small number of fundamental acoustic units to cover the acoustic variation in the sound space. However a large J will result in a feature vector with a very high dimension if we would like to cover as many unit combinations when forming acoustic words. For example, with a moderate value of $J = 256$, we have $M = 65\,792$ even when we only consider words less than or equal to two letters. This is already a very large dimensionality not commonly utilized in speech and language processing algorithms. Finally, a VSC-based classifier evaluates a goodness of fit, or score function $S_l(\boldsymbol{v}) = S(\boldsymbol{v}|\lambda_l)$, between a given vector, $\boldsymbol{v}$, and a model of the l-th spoken language, λ_l, to make a decision. Any vector-based classifier can be used to design spoken language identification and verification systems.

In language identification, we assume that each language to be recognized by the system has been registered and known to the system in advance. Therefore, it is a closed-set test. The objective is to identify the language l, among a pool of L candidates, that has the closest match or the highest score to a given test sample. System performance is often measured by the recognition error rate. On the other hand, in language verification, also known as language detection, the test language may or may not be known in advance to the system. This is therefore an open-set test. The objective of a verification test is to confirm whether a test sample belongs to the language that it is claimed to be. In this case, the test sample is compared against the model of the claimed language to produce a verification score. A decision is then made based upon whether the score is above or below a threshold. Two types of errors, *false alarms* (or false positives) and *misdetections* (or false negatives), are thus resulted. A wide range of error pairs can be obtained depending on the operating verification threshold of the system. To facilitate comparisons between different systems, an equal error rate (EER), indicating the error when the two rates are the same, is usually reported as the system performance. The NIST language recognition evaluation (LRE) is a series of open evaluations for benchmarking the progress of ongoing technology (http://www.nist.gov/speech/tests/index.htm).

Three sets of issues to be addressed in designing a VSC-based language classification system include: (i) fundamental unit selection and modeling, (ii) extraction of a spoken document vector, which can be considered as a front-end design, and (iii) vector-based classifier learning, which is labeled as a back-end design. These will be discussed in detail in the following three sections, respectively. Issues related to the accuracy of tokenization, the discrimination of feature vectors, and the performance of language classifiers will also be discussed in the remainder of the chapter. By having various combinations of the front- and back-ends, we have the flexibility to design a collection of systems that cover a wide spectrum of system performances and computation complexities.

42.2 Unit Selection and Modeling

The first issue to be considered is selection and modeling of the universal set of fundamental acoustic units. Spoken languages, despite sounding different from one another, do share some basic similarities in their acoustic characteristics. In the following, we list the 10 most common words in English and Chinese:

- *the* ... *of* ... *to* ... *a* ... *and* ... *in* ... *that* ... *for* ... *one* ... *is* ... (English)
- *de* (的) ... *yi* (一) ... *he* (和) ... *zai* (在) ... *shi* (是) ... *le* (了) ... *bu* (不) ... *you* (有) ... *zhe* (这) ... *ge* (个) ... (Mandarin)

By coincidence, the top-ranked English word, ("*the*"), and the most common Chinese Mandarin word, ("*de*"), share a similar pronunciation, resulting in a similar unit transcription if a common collection of sound unit models are used to decode both English and Mandarin speech segments containing this pair of words. Nonetheless, we can rely on the co-occurrence statistics of those words to discriminate one language from another. As discussed in Sect. 42.1, a decoder or tokenizer is needed to convert spoken utterances into sequences of fundamental acoustic units specified in an acoustic inventory. We believe that units that are not linked to a particular phonetic definition can be more universal, and are therefore conceptually easier to adopt. Such acoustic units are thus highly desirable for universal language characterization, especially for rarely observed languages or languages

without an orthographic system or a well-documented phonetic dictionary.

A number of variants have been developed along these lines, which were referred to as language-independent acoustic phone models. Hazen reported using 87 phones from the Oregon Graduate Institute multilanguage telephone speech corpus (OGI-TS) corpus [42.9]. *Berkling* [42.10] explored the possibility of finding and using only those phones that best discriminate between language pairs. *Berkling* [42.11] and *Corredor-Ardoy* [42.12] used phone clustering algorithms to find a common set of phones for languages. However, these systems are constrained to operate only when a phonetically transcribed database is available. On a separate front, a general effort to circumvent the need for phonetic transcription can be traced back to *Lee*'s work [42.13] in acoustic segment models (ASMs) in which a collection of ASMs were used to characterize the fundamental set of speech units, and constructed in an unsupervised manner without any linguistic definition. Acoustic words were thus formed by decoding word examples into sequences of such acoustic units, and then utilized for medium-vocabulary isolated-word recognition. A similar ASM approach has recently been adopted for language identification [42.14]. We first discuss the grouping of phone sets from multiple phone inventories of different languages to form a universal collection of units and corresponding phone models.

42.2.1 Augmented Phoneme Inventory (API)

Attempts have been made to derive a universal collection of phones to cover all sounds described in an international phonetic inventory, e.g., the international phonetic alphabet or Worldbet [42.15]. This is a challenging endeavor in practice. Note that these sounds overlap considerably across languages. One possible approximation is to form a superset of phonemes from several languages. We call this set the augmented phoneme inventory (API). This idea has been explored in one way or another in many studies [42.9–11]. A good inventory needs to cover phonetically as many targeted languages as possible. This method can be effective when phonemes from all the spoken languages form a closed set as studied by *Hazen* [42.9]. Results from human perceptual experiments have also shown a similar effect whereby listeners' language identification performance improved by increasing their exposure to multiple languages [42.16].

The API-based tokenization was recently studied [42.17] by using a set of all 124 phones from English, Korean, and Mandarin, plus four noise units, and extrapolating them to the other nine languages in the NIST LRE tasks. This set of 128 units is referred to as API I, which is a proprietary phone set defined for an internal database called the Institute for Infocomm Research language identification (IIR-LID) database. Many preliminary experiments were conducted using the IIR-LID database and the API I phone set. For example, we explored an API-based approach to universal language characterization [42.17], and a text categorization approach [42.7], which formed the basis for the vector-based feature extraction to be discussed in the next section. To expand the acoustic and phonetic coverage, we used another larger set of APIs with 258 phones, from the six languages defined by the OGI-TS database. These six languages all appear in the NIST LRE tasks. This set will be referred to as API II. A detailed breakdown of how the two phone sets were formed with phone set counts for each language is listed in Table 42.1.

Table 42.1 The languages and phone sets of API I and II

API I	Count	API II	Count
English	44	English	48
Mandarin	43	Mandarin	39
Korean	37	German	52
General	4	Hindi	51
		Japanese	32
		Spanish	36
Total	128	Total	258

Once the set of units are defined, models for each individual language can be obtained using conventional hidden Markov model (HMM) [42.18] tools trained on the speech examples specifically collected for the particular language. The collection of these phone models can be used to decode a given utterance by performing parallel phone recognition (PPR) [42.1]. The resulting multiple sequences of phone units can be used to form the spoken document vector corresponding to the given utterance. We will discuss both PPR and spoken document vectorization in later sections.

42.2.2 Acoustic Segment Model (ASM)

The conventional phone-based language characterization commonly used in automatic speech recognition and spoken language identification suffers from two major shortcomings. First, a combined phone set, such as API I mentioned above, from a limited set of multiple languages cannot be easily extended to cover new and rarely observed languages. Second, to train the acoustic mod-

els for each language, a collection of transcribed speech data is needed, but is often difficult to come by for all languages of interest. Furthermore, acoustic mismatches may exist among speech collected in different countries and under various acoustic conditions so that the collection of phone models may not work well for some intended languages under diverse test environments. To alleviate these difficulties, a data-driven method that does not rely on phonetically transcribed data is often preferred. This can be accomplished by constructing consistent acoustic segment models [42.13] intended to cover the entire sound space of all spoken languages in an unsupervised manner using the entire collection of speech training examples for all languages. The ASM framework takes advantage of the concept of language-independent acoustic phone models, and benefits from the unsupervised acoustic modeling technique.

Next, we present an ASM method to establish a universal representation of acoustic units for multiple languages [42.17]. As in any other hidden Markov modeling approaches, the initialization of ASMs is a critical factor to the success of ASM-based systems. Note that the unsupervised, data-driven procedure to obtain ASMs may result in many unnecessary small segments because of the lack of phonetic or prosodic constraints (e.g., the number of segments in a word and the duration of an ASM) during segmentation. This is especially severe in the case of segmenting a huge collection of speech utterances given by a large population of speakers from different language backgrounds. The API approach uses phonetically defined units in the sound inventory. It has the advantage of having phonetic constraints in the segmentation process. By using the API to bootstrap ASMs, we effectively incorporate some phonetic knowledge relating to a few languages in the initialization to guide the ASM training process, which is described as follows.

Step 1
Carefully select a few languages, typically with large amounts of labeled data, and train language-specific phone models. Choose a set of J models for bootstrapping.

Step 2
Use these J models to decode all training utterances in the training corpora. Assume the recognized sequences are *true* labels.

Step 3
Force-align and segment all utterances in the training corpora, using the available set of labels and HMMs.

Step 4
Group all segments corresponding to a specific label into a class. Use these segments to retrain an HMM.

Step 5
Repeat steps 2–4 several times until convergence.

In this procedure, we jointly optimize the J models as well as the segmentation of all utterances. This is equivalent to the commonly adopted segmental maximum likelihood (ML) and k-means HMM training algorithm [42.18] through iterative optimization of segmentation and maximization. We found that API-bootstrapped ASMs are more stable than the randomly initialized ASMs. There are also other ways of initializing ASMs, by imposing language-independent phonetic or prosodic constraints. The API-bootstrapped ASMs outperforms the API by a big margin in our 1996 NIST LRE task. Readers are referred to [42.17] for details of these ASM experiments. Using this single set of ASMs, we can now transcribe all spoken utterances from any language into sequences of units coming from a common alphabet. We call this process universal voice tokenization (UVT). Instead of using a single set of universal ASMs, we can also utilize multiple sets of fundamental units for parallel voice tokenization (PVT). For example, starting with the three-language API I units defined above, we can train three sets of language-dependent ASMs, bootstrapped from the phone models for each of the three languages. In this way, we can generate multiple sets of spoken documents, one from each language-specific tokenizer.

42.2.3 Comparison of Unit Selection

With an established acoustic inventory obtained from the API or the ASM method, we are now able to tokenize any given speech utterance into a token sequence $T(X)$, in a form similar to a text-like document. Note that ASMs are trained in a self-organized manner. We may not be able to establish a phonetic lexicon using ASMs and translate an ASM sequence into words. However, as far as language classification is concerned, we are more interested in a consistent tokenization than the underlying lexical characterization of a spoken utterance. The self-organizing ASM modeling approach offers a key advantage: we no longer require the training speech data to be phonetically transcribed.

In summary, the main difference between the API and ASM methods is the relaxation of phone transcription for segmentation. In the API approach, we train the phone models according to manually transcribed phone

labels while via ASMs, the segmentation is done in iterations using automatic recognition results. In this way, ASM gives us two advantages: (i) it allows us to adjust a set of API phones from a small number of selected languages towards a larger set of targeted languages; (ii) the ASM can be trained on the similar acoustic data as are used for the intended task, thus potentially minimizing mismatch between the test data and the API that is trained on a prior set of phonetically transcribed speech.

42.3 Front-End: Voice Tokenization and Spoken Document Vectorization

We are now ready to represent a spoken document or a spoken query by a vector whose dimensionality is equal to the size of the total number of useful features, including the statistics of the units and their co-occurrences. It is precisely with the usage of such high-dimensional vectors that we expect the discrimination capability of the feature vector to improve the performance even when only phonotactic features are used. The voice tokenization (VT) discussed here can typically be accomplished in a way similar to continuous phone recognition to decode a spoken utterance into a sequence of phone units. Readers are referred to [42.3] for an in-depth discussion of the modeling and decoding techniques.

Figure 42.1a,b illustrates a vectorization process that converts a spoken utterance into a document vector with the PVT and UVT front-ends. An unknown testing utterance is represented as a query vector by the front-end. For the PVT front-end with an F-language superset of APIs as initialization, the VSC vectorization (Fig. 42.1a) forms a large composite document vector, or supervector, $\boldsymbol{v} = [\boldsymbol{v}_1^{\mathrm{T}}, \ldots, \boldsymbol{v}_f^{\mathrm{T}}, \ldots, \boldsymbol{v}_F^{\mathrm{T}}]^{\mathrm{T}}$ by stacking F vectors, with each vector, $\boldsymbol{v}_f$, representing the document vector of the f-th language, obtained from the individual phone recognizers. On the other hand, for the UVT front-end, it constructs a single document vector $\boldsymbol{v}$ from the single phone recognizer. Figure 42.2a,b illustrate the language identification and verification processes using a four-language API superset example.

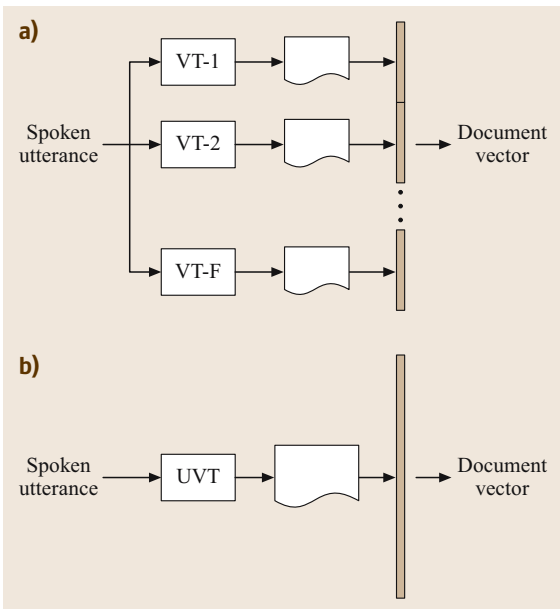


Fig. 42.1a,b Front-end: spoken utterance tokenization and vectorization **(a)** Spoken utterance tokenization and vectorization with PVT. **(b)** Spoken utterance tokenization and vectorization with UVT.

Since high-performance language-dependent acoustic and language models of phones are likely to be unnecessary for tokenization, we no longer need large amounts of training speech samples from each of the spoken languages. Instead, we only require a moderate-sized language-dependent training set of spoken documents in order to obtain a collection of spoken document vectors to be used to train a language-specific classifier for each language. When trigrams with 128 ASM units are incorporated, the vector dimensionality increases to over two million, which is well beyond the capability of current technology. It would be useful to find a balance between the *acoustic resolution*, i. e., the number of units J, needed to model the universal sound space for all languages, and the *language resolution*, i. e., the dimensionality of the spoken document vector M, needed to provide an adequate discriminative power for language identification. Readers are referred to [42.19] for a detailed study.

Many studies on vector-based document representation are available in information retrieval and text categorization literature [42.4, 5, 7, 20, 21]. In this chapter, we focus on language feature characterization that is used to discriminate between languages. Intuitively, sounds are heavily shared across different spoken languages because of the same human speech production mechanism that is used to produce them. The acoustic unit as proposed in Sect. 42.2 allows us to move away

from the conventional lexical descriptions of spoken languages. To account for the sequential, acoustic–phonotactic constraints, such as lexical constraints, we introduce the concept of an acoustic word (AW). An AW is typically smaller than a lexical word, and is composed of acoustic letters, such as the set of acoustic segment units, in an n-gram form. By exploiting the co-occurrence statistics of AWs, we can improve the discrimination power of the document vectors by incorporating acoustic words of different orders.

Suppose that we have a token sequence, $\{t_1, t_2, t_3, t_4\}$. We first derive the unigram statistics from the token sequence itself. We then compute the bigram statistics from $(\#t_1)$, (t_1t_2), (t_2t_3), (t_3t_4) and $(t_4\#)$ where the acoustic vocabulary is expanded to the token's right context. Similarly, we can extend this to the trigram statistics using $(\#t_1t_2)$, $(t_1t_2t_3)$, $(t_2t_3t_4)$ and $(t_3t_4\#)$ to account for both left and right contexts. The # sign is a place holder for free (or *do not care*) context. In the interest of manageability, we use only up to token trigrams. In this way, for an acoustic vocabulary of J tokens, we have potentially $J \times J$ bigram and $J \times J \times J$ trigram AWs to form a vocabulary of $M = J + J \times J + J \times J \times J$ AWs. Let us use the API I set defined in Sect. 42.2.1 to illustrate the concept of PVT-based vectorization. As shown in the left half of Table 42.1 that there are three subsets of units of sizes 44, 43, and 37 for English, Mandarin, and Korean, respectively. If we form AWs of up to two letters, this results in a lexicon of size $44^2 + 43^2 + 37^2 + 44 + 43 + 37 = 5278$, which is also the dimensionality of the document vectors.

This *bag-of-sounds* concept [42.22] is analogous to the *bag-of-words* paradigm originally formulated in the context of information retrieval and text categorization. In human languages, some words invariably occur more frequently than others. One of the most common ways to express this notion is known as the Zipf's law [42.23,24], which states that there is always a set of words that dominates most of the other words of a language in terms of their frequency of use. From some preliminary data, the theory can be extended to spoken words as well. This motivates the VSC-based approach.

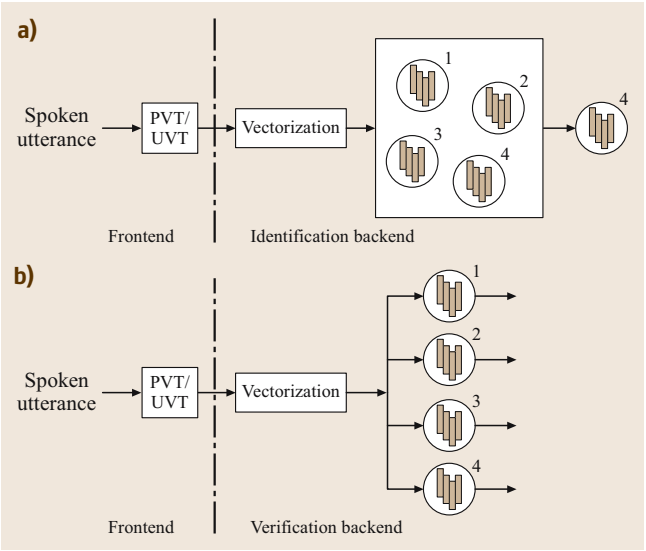


Fig. 42.2a,b Back-end: spoken language classification **(a)** Language identification using multiclass classification (four languages). **(b)** Language verification with independent Bayes decisions (four languages).

42.4 Back-End: Vector-Based Classifier Design

After a spoken utterance is tokenized and vectorized into a high-dimensional vector, language classification can be cast as a vector-based classification problem. There are many ways to construct the VSC back-end. The support vector machine (SVM) framework is a popular choice in many applications [42.5, 25]. Since the training of SVMs is usually optimized on a structural risk-minimization principle [42.26], SVMs have been shown to achieve good performances in several real-world tasks. In the rest of the chapter, we will use the SVM framework to illustrate the design of language classification systems. Other vector-based classifiers can also be utilized. Spoken language identification and verification system block diagrams for a four-language ($L = 4$) case are illustrated in Figs. 42.2a and b, respectively.

To train a two-class SVM, also known as a binary SVM, any input document vector $\boldsymbol{v}$ is labeled as Y_+ or Y_-, depending on whether the input belongs to the desired language category or not. Given a training database of D spoken document vectors, the binary SVM is a classifier of the form $g(\boldsymbol{v}) = \Omega^{\mathrm{T}}\psi(\boldsymbol{v}) + \omega_0$, characterized by a weight vector Ω, an offset ω_0, and a kernel function $\psi(\cdot)$. We can augment Ω with ω_0 and $\psi(\boldsymbol{v})$ with a unity, thus eliminating ω_0 in the rest of our discussion. SVM learning is usually posed as

an optimization problem with the goal of maximizing a margin, i. e., the distance between the separating hyperplane, $\Omega^{\mathrm{T}} \cdot \psi(\boldsymbol{v}) = 0$, and the nearest training vectors, such that if $g(\boldsymbol{v}) > 0$, then $\boldsymbol{v} \in Y_+$, and if $g(\boldsymbol{v}) \leq 0$, then $\boldsymbol{v} \in Y_-$. An extension of this formulation also allows for a wider margin at the cost of misclassifying some of the training examples. We used the SVM$^{\mathrm{light}}$ version 6.01 program to train all SVM models discussed in this chapter (http://svmlight.joachims.org/). This package allows us to explore both linear and nonlinear SVM kernels. A linear kernel SVM has $\psi(\boldsymbol{v}) = \boldsymbol{v}$. Other forms of kernels can also be used. Without loss of generality, we limit our discussion to linear-kernel SVM in the rest of this chapter.

42.4.1 Ensemble Classifier Design

When the number of classes L is greater than two, maximum margin classifier designs do not extend easily. Various approaches have been suggested. Some follow the concept of ensemble classifiers by having a collection of binary SVMs, such as *one-versus-rest*, *one-versus-one*, and their variants [42.27]; others build decision trees using top-down greedy algorithms or bottom-up clustering algorithms. We discuss these in detail in the present section.

Given L competing classes with all training vectors labeled as one of the L classes, the training set, T, is divided into a collection of L subsets, $T = \{T_1 \ldots, T_l \ldots, T_L\}$, with all samples that are labeled as class l belong to the set T_l.

One-Versus-Rest (OVR) SVM

This is conceptually the simplest multiclass ensemble classifier. We build L binary SVMs, each with the target l-th class as the positive category, with positive training set T_l, and the rest belonging to the negative category, with training set $\bar{T}_l = \{\boldsymbol{v}_i | i \neq l\}$ of negative examples. This approach is usually computationally expensive since we need to solve L quadratic program (QP) optimization problems, each involving D vectors with a large dimensionality, M. Given a test sample, the OVR decision function chooses the class that corresponds to the maximum value specified by the furthest hyperplane. As illustrated in Fig. 42.3a, the QP solutions result in the dotted line partitions while the decisions are made based on the solid line partitions. This gives errors as shown in the shaded area of Fig. 42.3a. Since there are usually more negative samples than positive ones, the OVR SVMs are usually hard to train because of the imbalance in the data distribution.

One-Versus-One (OVO) SVM

This method constructs binary SVMs for all $L \times (L-1)/2$ pairs of classes, one SVM for each pair of competing classes. When compared with OVR SVM, the OVO SVM method significantly reduces the computational needs in training because there are less training examples to train each pairwise SVM. In total, we need to solve $L \times (L-1)/2$ QP optimization problems. For each SVM, only data from the two competing classes are involved. Hence, we deal with a much smaller QP problem than that in OVR SVM. However, OVO SVM still leaves some undecided regions as shown in the shaded area of Fig. 42.3b.

Generalized SVM

This method is similar to OVR except that it modifies the support vector loss function to deal directly with multiple classes [42.28, 29]. Instead of solving a collection of independent SVMs, this method seeks to solve a single, monolithic QP problem of size $(L-1) \times D$, that

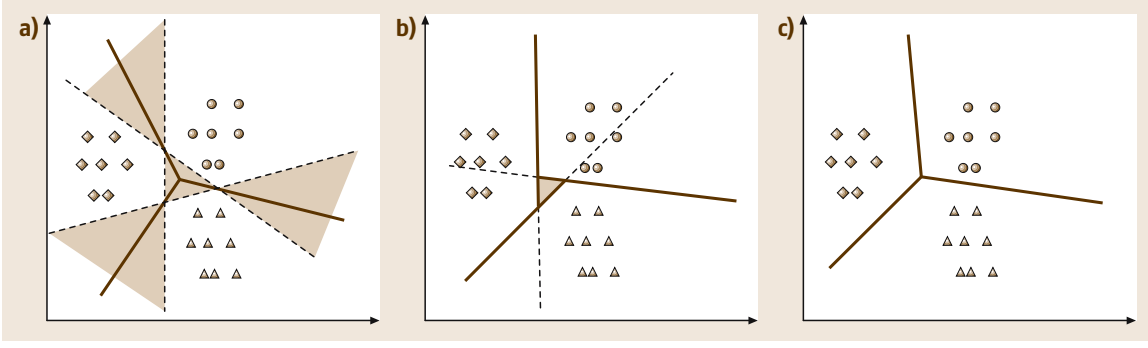


Fig. 42.3a–c SVM-based ensemble classifier design **(a)** One-versus-rest SVM. **(b)** One-versus-one SVM. **(c)** Generalized SVM.

maximizes the margins between all class pairs simultaneously. In general, it is computationally more expensive to solve a monolithic problem than multiple binary subproblems, such as is done by OVO SVMs, with the same amount of training data [42.25]. Generalized SVM can lead to L decision regions as illustrated in Fig. 42.3c. Although generalized SVMs are difficult to learn, other multiclass classifiers can serve this purpose. For example a maximum figure-of-merit (MFoM) learning algorithm has been shown to be effective in discriminative multiclass text categorization [42.6] and spoken language identification [42.7].

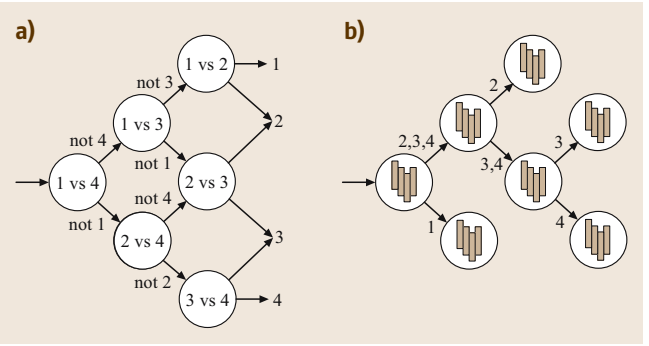


Fig. 42.4a,b Decision strategy **(a)** Direct acyclic graph. **(b)** Binary decision tree.

42.4.2 Ensemble Decision Strategy

The idea of an ensemble decision is to consult a large number of independently constructed classifiers to make a decision based on consensus. It provides a solution to the multiclass classification problem with independently trained binary classifiers. First, we construct many subordinate classifiers, each responsible for removing some uncertainty regarding the class assignment of the test sample; and secondly, we apply classification schemes to make collective decisions. To make a final decision for a multiclass classification problem with an ensemble of binary classifiers, two strategies have been well studied, the maximum wins (max. wins) (MW) and directed acyclic graph (DAG) strategies. The MW strategy makes a collective decision based on a number of individual decisions. Each decision is represented by an individual binary SVM. The collection of individual SVMs is arranged in a flat structure. In practice, we evaluate a test sample against all the $L\times(L-1)/2$ binary SVMs. The class that gains most of the winning votes is then chosen as the identified class.

A DAG is a graph whose edges have an orientation but with no cycles. The DAG strategy [42.30] uses a rooted binary directed acyclic graph which has $L\times(L-1)/2$ internal nodes with L leaves. Figure 42.4a gives an example of a four-class DAG classifier, with six internal nodes and four leaves. Each node represents one of the $L\times(L-1)/2$ binary SVM. To evaluate a test sample $\boldsymbol{v}$, starting at the root node, a binary decision is made at each node to eliminate one candidate class every time a node is visited. The node that survives all the eliminations is eventually chosen as the identified class. Readers can walk through the example in Fig. 42.4a. Thus, for a problem of L-class classification, there are $L-1$ steps of decisions before we arrive at an answer. Furthermore, a node or leaf in DAG is reachable by more than one possible path through the system. Therefore, the decision graph that the path traverses is a DAG, and not simply a tree. In this way, DAG allows for a more efficient representation of redundancies and repetitions that can occur in different branches of the tree, by allowing possible merging of different decision paths. The difference between a DAG and a decision tree can be observed by comparing the tree structures in Fig. 42.4b and the DAG in Fig. 42.4a.

Although both the MW and DAG strategies make decisions based on the same OVO SVMs, the DAG approach demonstrates two clear advantages: (i) with a decision tree architecture, DAG is a much more efficient strategy than the MW during testing. It only involves $(L-1)$ SVM operations, as opposed to $L\times(L-1)/2$ comparisons in MW; and (ii) DAG is amenable to a VC-style bound on the generalization error while bounds on the generalization errors have yet to be establish for the MW method [42.30].

While the MW and DAG strategies can produce accurate class predictions, they lack a structural representation for class similarities. With a large number of classes, we may want to have a meaningful organization of these classes. As shown in Fig. 42.4b, a decision tree presents the classes in a hierarchical structure of clusters and subclusters according to their proximity. It makes a final decision through steps of subdecisions for a test sample. At each node in the tree, a decision is made to reduce the uncertainty of the test samples. Starting from the root node through the successor nodes, we make the final decision when a leaf node is reached.

42.4.3 Generalized VSC-Based Classification

We have discussed designing ensemble classifiers and decision strategies based on high-dimension feature vectors and binary SVMs. In some cases, it is beneficial to

reduce feature dimensions in order to compensate for the lack of training vectors and potential measurement noise in feature extraction. Once we have a vector of lower dimension many well-known probabilistic modeling and machine learning techniques can be used to design classifiers and formulate decision rules.

We also consider an indirect vectorization mechanism based on the scores obtained from the ensemble classifiers discussed earlier. Since these scores characterize the distribution of the outputs of a variety of classifiers, they can be grouped together to form a score *supervector* describing an overall behavior of a score distribution over different classifiers on all competing classes. One example of such a supervector is to concatenate HMM state scores from all competing models to form an overall score vector. This approach has been shown to have good discrimination power in isolated-word recognition [42.31, 32]. Since these score supervectors are usually obtained from a finite set of ensemble classifiers, their dimensionality is usually much smaller than that of the spoken document vectors discussed in Sect. 42.3. These score supervectors are more amendable to using the conventional probabilistic modeling approaches. We call this combination framework generalized VSC-based classification; it uses high-dimensional feature vectors to generate scores from a collection of ensemble classifiers, and forms low-dimensional score supervectors to perform final classifier design and decisions. All the language identification and verification experiments to be discussed in Sect. 42.5 are based on this generalized VSC classification approach.

Feature Reduction

In many cases, it is desirable to reduce the dimensionality of the document vectors to a manageable size so that probabilistic models, such as a Gaussian mixture model (GMM), can be easily utilized. Many dimensionality reduction approaches, such as truncated singular value decomposition (SVD) [42.8] or principal component analysis (PCA) [42.33], have been studied. Suppose that we are given a database with D documents and M distinguished attributes, also known as terms. Let $\boldsymbol{A}$ denote the corresponding $M \times D$ document-term matrix with entries $a_{m,d}$ that represent the importance of the m-th term in the d-th document. SVD effectively reduces the dimensionality by finding the closest rank-R approximation to $\boldsymbol{A}$ in the Frobenius norm, while PCA finds the R-dimensional subspace that best represents the full data with respect to a minimum squared error. Although the SVD or the PCA method finds subspaces that are useful for representing the original high-dimensional vector space, there is no reason to assume that the resulting projections must be useful for discriminating between data in different classes [42.34].

Linear discriminant analysis finds a decision surface, also known as a hyperplane, that minimizes the *sample risk* or *misclassification error* for linearly separable classes. In the two-class case, the linear discriminant function [42.34] is expressed as $g(\boldsymbol{v}) = \Omega^{\mathrm{T}}\boldsymbol{v}$, with $g(\boldsymbol{v})$ representing the signed distance between $\boldsymbol{v}$ and the decision surface $\Omega^{\mathrm{T}}\boldsymbol{v} = 0$. In this way, from the perspective of dimensionality reduction, a multidimensional feature vector $\boldsymbol{v}$ is projected to a one-dimensional space by $g(\boldsymbol{v})$.

Vectorization with Ensemble Scores

If we employ a linear SVM for each of the subordinate classifiers, the outputs $g(\boldsymbol{v})$ from the SVMs then form a vector of signed distance. In this way, a high-dimensional document vector can be effectively reduced to a much lower dimensionality. From an OVR SVM ensemble classifier, we obtain a vector of $Q = L$ dimensionality; from an OVO SVM ensemble classifier, we arrive at a vector of $Q = L \times (L-1)/2$ dimensionality. Clearly, not only do we effectively reduce the dimensionality from a large M to a small Q, but we also represent the spoken document vector in a discriminative space of language pairs.

Studies show that an ensemble classifier performs well when the number of subordinate classifiers Q is set to around $10 \log_2 L$. In general, more subordinate classifiers improve performance, but at a higher computational cost [42.35]. Hence, from the perspective of expressiveness, OVO SVM ensemble classifier seems to be a better choice than OVR SVM when we are dealing with 15 languages as in *CallFriend*, where we have $105 = 15 \times (15-1)/2$ OVO SVM outputs versus 15 OVR SVM outputs (http://www.ldc.upenn.edu/Catalog/project_index.jsp).

Generalized Vector-Based Classifiers and Decision Rules

We mentioned above that generalized SVMs attempt to design ensemble classifiers by considering all possible partitions of the vector space as shown in Fig. 42.3c, which is usually a difficult problem. On the other hand, if we can define a class discriminant function, $S(\boldsymbol{v}|\lambda_l)$ for the l-th class with λ_l as the model for the target l-th class, and be able to consistently rank a goodness-of-fit score between the unknown vector $\boldsymbol{v}$ and the model λ_l, a decision rule for identification can easily be formulated as:

$$\hat{l} = \underset{l}{\operatorname{argmax}}\, S(\boldsymbol{v}|\lambda_l)\,. \tag{42.1}$$

Similarly, a verification decision can be made to accept the claimed class identity l if:

$$S(\boldsymbol{v}|\lambda_l) - S(\boldsymbol{v}|\bar{\lambda}_l) > \tau_l\,, \tag{42.2}$$

with $S(\boldsymbol{v}|\bar{\lambda}_l)$ denoting an antidiscriminant function score that is typically a score representing the collective contribution from all competing classes other than the l-th class, and τ_l denoting the verification threshold for the l-th class.

One good example is a linear discriminant function (LDF)-based score function $S(\boldsymbol{v}|\Omega_l) = \Omega_l{}^{\mathrm{T}}\boldsymbol{v}$ mentioned early in the SVM design, which is simply an inner product between the weight vector Ω_l of the l-th class and the unknown vector $\boldsymbol{v}$. Another commonly used example is to model each class by a GMM, so that we have $S(\boldsymbol{v}|\lambda_l)$, evaluating the likelihood of observing $\boldsymbol{v}$ under GMM λ_l. Artificial neural networks (ANNs) [42.36] are another popular approach that can be used to approximately map the input vectors to the output decisions. Results of using both LDF- and ANN-based generalized classifier design approaches have been recently reported in [42.37] for spoken language classification in the 2005 NIST LRE task. Classifier learning is based on the minimum classification error (MCE) and maximal figure-of-merit (MFoM) training principles which have been extensively studied in and successfully applied to speech recognition [42.38], and text categorization [42.6] tasks.

42.5 Language Classification Experiments and Discussion

We illustrate the VSC framework for the language identification results on the primary subset of 30 s test segments of the 1996 NIST LRE task. On the other hand, language verification benchmarks are established based on the 1996 and 2003 NIST LRE tasks.

42.5.1 Experimental Setup

The training sets for building models came from three sets of corpora, namely: (i) the three-language IIR-LID database [42.7] with English, Mandarin, and Korean; (ii) the six-language OGI-TS (multilanguage telephone speech, http://cslu.cse.ogi.edu/corpora/corpCurrent.html) database with English, German, Hindi, Japanese, Mandarin, and Spanish; and (iii) the 12-language Linguistic Data Consortium (LDC) *CallFriend* database. The overlap between the *CallFriend* database and the 1996 LRE data was removed from the training data as suggested http://www.nist.gov/speech/tests/index.htm for 2003 LRE. As English, Mandarin, and Spanish each have two accented versions in the *CallFriend* database, in all, we were given a 15-language set, which includes the three additional accents. The IIR-LID and OGI-TS databases were only used for bootstrapping the acoustic models as an initial set of phones. Both IIR-LID and OGI-TS databases are telephone speech with phonetic transcriptions. In addition, the *CallFriend* database was used for fully fledged ASM acoustic modeling, vectorization, and classifier design. It contains telephone conversations of the same 12 languages that are in the 1996 and 2003 NIST LRE tasks, but without phonetic transcriptions. The three databases were recorded independently of each other.

In the IIR-LID database, each language contains more than 150 hours of speech, while in the OGI-TS database, each language amounts to less than one hour of speech. Furthermore in the *CallFriend* database, each of the 15 language databases consists of 40 telephone conversations with each lasting approximately 30 min, giving a total of about 20 hours per language. For vectorization and classifier design, each conversation in the training set was segmented into overlapping sessions, resulting in about 12 000 sessions for each duration per language. For testing, there were three different duration settings: 3, 10, and 30 s. The 1996 NIST LRE evaluation data consist of 1503, 1501, and 1492 sessions of speech segments with 3, 10, and 30 s in length, respectively. The 2003 NIST LRE evaluation data consist of 1280 sessions per duration. The test data were labeled with the 12 main languages only, without considering their accent type.

We configured each ASM with a three-state left-to-right hidden Markov model. It was found that a 32-mixture Gaussian state density provides adequate results as compared with a 16- or 8-mixture Gaussian state [42.17]. It was also reported that the ASM set derived from API II (258-ASM) slightly outperforms that from API I (128-ASM) by having a broader acoustic coverage [42.19]. We will adopt the 128-ASM for language identification and the 258-ASM for language verification experiments.

Table 42.2 Error rates (ER%) and number of support vectorson 30 s NIST 1996 LRE

No. sessions per language	1000	2000	6000	12 000	Testing cost (no. SVM decisions)
DAG (ER%)	18.2	16.2	14.4	13.9	$L-1$
Max. wins (ER%)	19.0	16.7	15.1	14.5	$L\times(L-1)/2$
No. support vectors	1048	1457	1951	2142	

With 15 languages including accents in the *CallFriend* database, we trained $105 = 15\times(15-1)/2$ OVO SVMs to model the languages and accents. In language identification, we implemented both MW and DAG decision strategies based on OVO SVMs. The conventional language model (LM) scoring techniques used in the parallel phone recognition followed by language modeling (P-PRLM) approaches [42.1] can also be used as a back-end to contrast the VSC back-end discussed earlier. For language verification, to appreciate the contributions of the different front- and back-ends, we construct and test four combination systems using the PVT and UVT front-ends, and the VSC and LM back-ends. In the 1996 NIST LRE task, we built a system to identify 12 languages.

42.5.2 Language Identification

For language identification, the objective is to identify the most likely language given a test sample, as illustrated in Fig. 42.2a. This is usually considered as a closed-set problem assuming that all the languages to be decided are known to the system in advance. We follow the experiment setup in the NIST LRE tasks, and the evaluation is carried out on recorded telephony speech of 12 languages for the 1996 and 2003 NIST LRE: Arabic, English, Farsi, French, German, Hindi, Japanese, Korean, Mandarin, Spanish, Tamil, and Vietnamese.

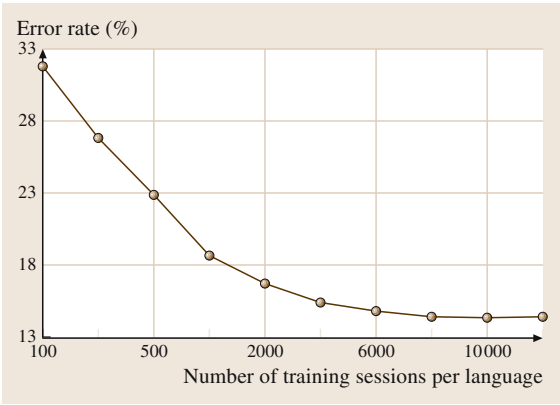


Fig. 42.5 Error rates (ER%) as a function of training set size evaluated on 30 s NIST 1996 LRE

It is well known that the size of the training set often affects the performance of a pattern classification system significantly. Let us study the effects of training set size on system performance using the DAG strategy. We proceed with the document vectors based on bigram AWs derived from the 128-ASM set. In Fig. 42.5, we reported the language identification error rate as a function of the number of training sessions (vectors). As the number of training sessions increase, the training cost of SVMs increase as well. The full corpus includes 12 000 spoken documents, or sessions, for each language/accent. The subset was randomly selected from the full corpus with an equal amount of data from each language. We observe that, when subset size grows beyond 8000 sessions per language, performance begins to saturate.

In Table 42.2, we compare the error rates using different training corpus sizes evaluated on the 30 s NIST 1996 LRE test set. We also report the number of resulting support vectors in the set of binary SVMs. When the number of sessions per language was 1000, there were 2000 training vectors for each language pair. The two-class SVMs gave about 1048 support vectors per binary SVM on average. Note that the number of support vectors increases as training corpus grows, but at a much slower rate than that of the training corpus size. When the training corpus size grows by 12 times, the number of support vectors only doubles. This explains the fact that, beyond 8000 sessions, increasing the training corpus size does not translate into substantial accuracy improvements.

As discussed in Sect. 42.4.1, both the MW and DAG SVM strategies make decisions based on the same OVO SVMs. We further compare the performance of the two strategies in Table 42.2. It is worth pointing out that it is much less expensive to train a set of OVO SVMs than a set of OVR SVMs [42.30]. From the perspective of computation cost in testing, the DAG strategy involves $L/2$ times less computation than MW does. The DAG approach also demonstrates superior performance to that obtained from the MW strategy in terms of accuracy consistency on all tests.

42.5.3 Language Verification

In the case of language verification, the goal is to verify whether a language identity claim is true or false. To this end, a typical VSC back-end consists of multiple independent Bayes decisions, each making a decision for a candidate language, as in Fig. 42.2b. Language verification is an open-set hypothesis testing problem. The language verification experiments discussed next follow the procedure in Sect. 42.4.3. We have described the two different front-ends, PVT and UVT, with both the VSC and LM back-ends. To gain further insight into the behavior of each of the front- and back-ends, it is desirable to investigate the performance of each of the four system combinations, namely PVT–LM, PVT–VSC, UVT–LM, and UVT–VSC, where the PVT/UVT front-ends are built on a set of universal ASMs.

Without loss of generality, we deployed the same 258-ASM in two different settings. First, the 258 ASMs were arranged in a six-language PVT front-end. They were redistributed according to their API II definitions into three languages. Second, they were lumped together in a single UVT front-end. The training of 258-ASM was discussed in Sect. 42.2.2.

The PVT/UVT front-end converts spoken documents into high-dimensional vectors as illustrated in Fig. 42.2a,b. The six-language PVT front-end generates vectors of 11 708 ($=48^2+39^2+52^2+51^2+32^2+36^2+48+39+52+51+32+36$) dimensions (PVT vectors), while the UVT front-end generates those of 66 822 $\left(=258^2+258\right)$ dimensions (UVT vectors).

We can use the vectors generated by the PVT and UVT front-ends directly. We can also use the scores generated by the binary classifiers. With the OVO SVMs, we further convert the document vectors into $105=15\times(15-1)/2$ attributes, with each attribute representing a signed distance between a document vector and the decision hyperplane of an OVO SVM. This score supervector represents 99.1% and 99.8% dimensionality reductions from the original PVT and UVT vector dimensions, respectively. We further train 512-mixture GMMs, λ^{x+} and λ^{x-}, for each of the languages, and report the equal error rates (EER in percentage) between misdetections and false-alarms.

The UVT–LM system follows the block diagram of the language-independent acoustic phone recognition approach [42.12]. The PVT–LM is implemented as in [42.1]. The LM back-end uses trigrams to derive phonotactic scores. The results on the 1996 and 2003 NIST LRE tasks are reported in Tables 42.3 and 42.4, respectively.

Table 42.3 Equal error rates (ERR%) comparison of four systems on NIST 1996 LRE

System	30 s	10 s	3 s
PVT–VSC	2.75	8.23	21.16
PVT–LM	2.92	8.39	18.61
UVT–VSC	4.87	11.18	22.38
UVT–LM	6.78	15.90	27.20

Table 42.4 Equal error rates (ERR%) comparison of four systems on NIST 2003 LRE

System	30 s	10 s	3 s
PVT–VSC	4.02	10.97	21.66
PVT–LM	4.62	11.30	21.18
UVT–VSC	6.81	13.75	24.44
UVT–LM	10.81	19.95	30.48

Before looking into the results, let us examine the combinative effect of the front- and back-ends. In these combinative systems, there are two unique front-end settings, PVT and UVT. The PVT converts an input spoken utterance into six spoken documents using the parallel front-end, while the UVT converts an input into a single document. The LM in the PVT–LM and that in the UVT–LM are different; the former has 6×12 n-gram language models while the latter has only 12 language models. That is to say, the former LM classifier is more complex, with a larger number of parameters. On the other hand, the VSC in the PVT–VSC and the VSC in the UVT–VSC are of different complexities as well. Although the dimensionality of the supervector from PVT is lower than that of UVT, the supervector is several times as dense as that of UVT because there are many low-occurrence AWs in the case of UVT, resulting in more-complex SVMs [42.26]. In other words, the VSC classifier in the PVT–VSC is more complex than that in UVT–VSC. In terms of the overall classifier back-end complexity, we rank the four systems from high to low as follows: PVT–VSC, PVT–LM or UVT–VSC, and UVT–LM.

We now summarize what we have discussed: (i) the VSC back-end demonstrates a clear advantage over the LM back-end for the 30 s trials, while LM works better for the 3 s trials in general. This can easily be explained by the fact that the VSC models are designed to capture higher-order phonotactics. As a result, VSC favors long utterances which provide a richer set of long span phonotactic information over short utterances; (ii) the system performance correlates highly with the complexity of system architectures. This can be found

consistently in Tables 42.3 and 42.4, where the PVT–VSC presents the best result with an EER of 2.75% and 4.02% for the 30 s 1996 and 2003 NIST LRE tasks, respectively. It is then followed by PVT–LM, UVT–VSC, and UVT–LM. Note that we can increase the system complexity by including more phone recognizers in PVT. We expect more phone recognizers to further improve the PVT–VSC system performance. As a general remark, the OVO SVM followed by the DAG decision strategy is shown to be computationally effective in both training and testing in language identification. The OVO SVM also delivers outstanding system performances in both identification and verification tasks.

42.5.4 Overall Performance Comparison

We summarize the overall performance of the SVM approach and compare its results with other state-of-the-art systems recently reported in the literature. Only the results of the 30 s segment testing, which represent the primary interest in the LRE tasks, are listed in Table 42.5. Here systems 1–3 are trained and tested on the same database configuration. Therefore, the results can be directly compared. We extract the corresponding results from Tables 42.3 and 42.4. Two sets of recently reported results are worth mentioning. They are listed under systems 4 and 5, and extracted from [42.39] and [42.40], respectively. It is clear from the results in Table 42.5 that the performance of the PVT–VSC system represents one of the best reported results on the 1996 and 2003 NIST LRE tasks.

Table 42.5 Equal error rates (ERR%) benchmark on 30 s NIST 1996/2003 LRE (in [42.40], Russian data are excluded)

	System	1996 LRE	2003 LRE
1	PVT–VSC	2.75	4.02
2	PVT–LM	2.92	4.62
3	UVT–VSC	4.87	6.81
4	Phone lattice [42.39]	3.20	4.00
5	Parallel PRLM [42.40]	5.60	6.60

The proposed VSC-based language classifier only compares phonotactic statistics from spoken documents. We have not explored the use of the acoustic scores resulting from the tokenization process. It was reported that there is a clear win in combining information about the acoustic scores along with the phonotactic statistics [42.12, 40, 41]. Furthermore, the fusion of phonotactic statistics at different levels of resolutions also improves overall performance [42.42]. We have good reasons to expect that fusion among our four combinative systems, or between our systems and other existing methods including the acoustic score classifier [42.41] and GMM tokenizer [42.43], will produce further improvements. This has been demonstrated by some recently reported results, such as the 2.70% EER on the 2003 NIST LRE in [42.39]. However, it is beyond the scope of this chapter to discuss classifier fusions.

42.6 Summary

We have presented a vector-based spoken language classification framework that addresses a range of questions from the theoretical issue of the nature of the mind to more-practical aspects such as design considerations. As a case study, this chapter covers three areas: (i) a vector space characterization strategy for representing spoken documents; (ii) a systematic discussion of language identification and verification formulation cast in the discriminative classifier design strategy; (iii) a number of practical issues for system implementation in applications such as the NIST LRE. Using a conventional SVM classifier design with the PVT front-end and VSC back-end combination, we achieved an EER of 2.75% and 4.02% in the 30 s 1996 and 2003 NIST LRE tasks, respectively. This represents one of the best reported results using a single classifier.

We have studied a number of practical issues in ensemble classifier design, such as the computational needs in the training and testing of different strategies. In language identification, we have successfully demonstrated through a case study that DAG is a better solution than MW in terms of computational efficiency and system accuracy. Multiclass SVM by itself is still an ongoing research issue. Moving forward, recent progress in discriminative classifier designs, such as MFoM [42.6] and decomposition method [42.25] for monolithic solutions to multiclass problems, will result in further improvements. In language verification, we introduced the concept of using SVM decision hyperplanes as the projection directions for dimensionality reduction. This allows us to carry out language verification under the classical GMM framework. The use of

OVO SVM outputs can also be seen as an implementation of the concept of error-correcting output coding (ECOC) [42.44, 45], which makes classification decision by a consensus among subordinate classifiers. One of the central issues in ECOC is choosing a good code. It will be interesting to study other ECOC codes beyond OVO SVMs.

One of the important properties of vectorization is that it handles the fusion of different types of language cues in a high dimensional vector seamlessly. We have successfully implemented the *bag-of-sounds* spoken document vectors using a mix of n-gram statistics of *acoustic letters*. We believe that this framework can be extended to accommodate heterogeneous attributes such as acoustic and prosodic features as well. Whereas fusion of classifier outputs has been the focus of recent research [42.40, 46], vector space modeling explores a new horizon where fusion can take place at the feature level. The same classifier design theory is applicable to other vector-based representation such as GMM *supervector* [42.47], maximum-likelihood linear regression (MLLR) vector [42.48], and generalized linear discriminant sequence (GLDS) vector representation [42.49].

In summary, vector-based spoken language classification benefits from progress in the areas of acoustic modeling, machine learning, and text-based information retrieval. We have successfully treated spoken language classification as a text categorization problem with the topic category being the language identity itself. The same theory also applies to other spoken document classification tasks, such as topic detection and tracking, and *query-by-example* spoken document retrieval. This is another highly anticipated research direction.

References

42.1 M.A. Zissman: Comparison of four approaches to automatic language identification of telephone speech, IEEE Trans. Speech Audio Process. **4**(1), 31–44 (1996), 1

42.2 C.-H. Lee, F.K. Soong, K.K. Paliwal (Eds.): *Automatic Speech and Speaker Recognition: Advanced Topics* (Kluwer Academic, Dordrecht 1996)

42.3 J.L. Gauvain, L. Lamel: Large-vocabulary continuous speech recognition: advances and applications, Proc. IEEE **88**(8), 1181–1200 (2000)

42.4 G. Salton: *The SMART Retrieval System* (Prentice-Hall, Englewood Cliffs 1971)

42.5 F. Sebastiani: Machine learning in automated text categorization, ACM Comput. Surv. **34**(1), 1–47 (2002)

42.6 S. Gao, W. Wu, C.-H. Lee, T.-S. Chua: A MFoM learning approach to robust multiclass multi-label text categorization, Proc. ICML (2004) pp. 42–49

42.7 S. Gao, B. Ma, H. Li, C.-H. Lee: A text-categorization approach to spoken language identification, Proc. Interspeech (2005) pp. 2837–2840

42.8 J.R. Bellegarda: Exploiting latent semantic information in statistical language modeling, Proc. IEEE **88**(8), 1279–1296 (2000)

42.9 T.J. Hazen: *Automatic Language Identification Using a Segment-based Approach* (MIT, Cambridge 1993), MS Thesis

42.10 K.M. Berkling, E. Barnard: Analysis of phoneme-based features for language identification, Proc. ICASSP (1994) pp. 289–292

42.11 K.M. Berkling, E. Barnard: Language identification of six languages based on a common set of broad phonemes, Proc. ICSLP (1994) pp. 1891–1894

42.12 C. Corredor-Ardoy, J.L. Gauvain, M. Adda-Decker, L. Lamel: Language identification with language-independent acoustic models, Proc. Eurospeech, Vol. 1 (1997) pp. 55–58

42.13 C.-H. Lee, F.K. Soong, B.-H. Juang: A segment model based approach to speech recognition, Proc. ICASSP (1988) pp. 501–504

42.14 A.K.V.S. Jayram, V. Ramasubramanian, T.V. Sreenivas: Language identification using parallel subword recognition, Proc. ICASSP (2003) pp. 32–35

42.15 J.L. Hieronymus: ASCII phonetic symbols for the world's languages: Worldbet, Technical Report AT&T Bell Labs (1994)

42.16 Y.K. Muthusamy, N. Jain, R.A. Cole: Perceptual benchmarks for automatic language identification, Proc. ICASSP (1994) pp. 333–336

42.17 B. Ma, H. Li, C.-H. Lee: An acoustic segment modeling approach to automatic language identification, Proc. Interspeech (2005) pp. 2829–2832

42.18 L.R. Rabiner: A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE **77**(2), 257–286 (1989)

42.19 H. Li, B. Ma, C.-H. Lee: A vector space modeling approach to spoken language identification, IEEE Trans. Audio Speech Language Process. **15**(1), 271–284 (2007)

42.20 H.K.J. Kuo, C.-H. Lee: Discriminative training of natural language call routers, IEEE Trans. Speech Audio Process. **11**(1), 24–35 (2003)

42.21 J. Chu-Carroll, B. Carpenter: Vector-based natural languagecall routing, Comput. Linguist. **25**(3), 361–388 (1999)

42.22 H. Li, B. Ma: A phonotactic language model for spoken language identification, Proc. ACL (2005) pp. 515–522

42.23 G.K. Zipf: *Human Behavior and the Principal of Least Effort, An Introduction to Human Ecology* (Addison-Wesley, Reading 1949)

42.24 K.S. Jones: A statistical interpretation of term specificity and its application in retrieval, J. Doc. **28**, 11–20 (1972)

42.25 C.-W. Hsu, C.-J. Lin: A comparison of methods for multiclass support vector machines, IEEE T. Neural Netw. **13**(2), 415–425 (2002)

42.26 V. Vapnik: *The Nature of Statistical Learning Theory* (Springer, Berlin, Heidelberg 1995)

42.27 A. Statnikov, C. Aliferis, I. Tsamardinos, D. Hardin, S. Levy: A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis, Bioinformatics **21**(5), 631–643 (2005)

42.28 J. Weston, C. Watkins: Multi-class support vector machines, Tech. Rep. CSD-TR-98-04 (University of London, London 1998)

42.29 Y. Lee, Y. Lin, G. Wahba: Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data, J. Am. Stat. Assoc. **99**(465), 67–81 (2004)

42.30 J.C. Platt, N. Cristianini, J. Shawe-Taylor: Large margin DAG's for multiclass classification, Advances in Neural Information Processing Systems, Vol. 12 (Cambridge, MIT Press 2000) pp. 547–553

42.31 S. Katagiri, C.-H. Lee: A New Hybrid Algorithm for Speech Recognition Based on HMM Segmentation and Discriminative Classification, IEEE Trans. Speech Audio Process. **1**(4), 421–430 (1993)

42.32 K.-Y. Su, C.-H. Lee: Speech Recognition using Weighted HMM and Subspace Projection Approaches, IEEE Trans. Speech Audio Process. **2**(1), 69–79 (1994)

42.33 M. Kobayashi, M. Aono: Vector space models for search and cluster mining. In: *Survey of Text Mining*, ed. by M.W. Berry (Springer, Berlin, Heidelberg 2003)

42.34 R.O. Duda, P.E. Hart, D.G. Stork: *Pattern Classification* (Wiley, New York 2001)

42.35 K. Crammer, Y. Singer: Improved Output Coding for Classification Using Continuous Relaxation, Proc. NIPS (2000) pp. 437–443

42.36 S. Haykin: *Neural Networks: A Comprehensive Foundation* (McMillan, London 1994)

42.37 J. Li, S. Yaman, C.-H. Lee, B. Ma, R. Tong, D. Zhu, H. Li: Language recognition based on score distribution feature vectors and discriminative classfier fusion, Proc IEEE Odyssey Speaker and Language Reognition Workshop (2006)

42.38 S. Katagiri, B.-H. Juang, C.-H. Lee: Pattern Recognition Using A Generalized Probabilistic Descent Method, Proc. IEEE **86**(11), 2345–2373 (1998)

42.39 J.L. Gauvain, A. Messaoudi, H. Schwenk: Language recognition using phone lattices, Proc. ICSLP (2004) pp. 1215–1218

42.40 E. Singer, P.A. Torres-Carrasquillo, T.P. Gleason, W.M. Campbell, D.A. Reynolds: Acoustic, phonetic and discriminative approaches to automatic language recognition, Proc. Eurospeech (2003) pp. 1345–1348

42.41 P.A. Torres-Carassquilo, E. Singer, M.A. Kohler, R.J. Greene, D.A. Reynolds, J.R. Deller Jr.: Approaches to language identification using Gaussian mixture models and shifted delta cepstral features, Proc. ICSLP (2002) pp. 89–92

42.42 B.P. Lim, H. Li, B. Ma: Using local and global phonotactic features in Chinese dialect identification, Proc. ICASSP (2005) pp. 577–580

42.43 P.A. Torres-Carrasquillo, D.A. Reynolds, R.J. Deller Jr.: Language identification using Gaussian mixture model tokenization, Proc. ICASSP (2002) pp. 757–760

42.44 T.G. Dietterich, G. Bakiri: Solving multiclass learning problems via error-correcting output codes, J Artif. Intell. Res. **2**, 263–286 (1995)

42.45 H. Li, B. Ma, R. Tong: Vector-Based Spoken Language Recognition using Output Coding, Proc. Interspeech (2006)

42.46 R. Tong, B. Ma, D. Zhu, H. Li, E.S. Chng: Integrating acoustic, prosodic and phonotactic features for spoken language identification, Proc. ICASSP **1**, 205–208 (2006)

42.47 W.M. Campbell, D.E. Sturim, D.A. Reynolds: Support vector machines using GMM Supervectors for speaker recognition, IEEE Signal Process. Lett. **13**(5), 308–311 (2006)

42.48 A. Stolcke, L. Ferrer, S. Kajarekar, E. Shriberg, A. Venkataraman: MLLR transforms as features in speaker recognition, Proc. Interspeech (2005) pp. 2425–2428

42.49 W.M. Campbell, J.P. Campbell, D.A. Reynolds, E. Singer, P.A. Torres-Carrasquillo: Support vector machines for speaker and language recognition, Comput. Speech. Lang. **20**(2-3), 210–229 (2005)

49. Active Noise Control

S. M. Kuo, D. R. Morgan

This chapter introduces principles, algorithms, and applications of active noise control (ANC) systems. We emphasize the practical aspects of ANC systems in terms of adaptive algorithms for real-world applications. The basic algorithm for ANC is first developed and analyzed based on broadband feedforward control. This algorithm is then modified for narrowband feedforward and adaptive feedback ANC systems. Finally, these single-channel ANC algorithms are expanded to multichannel systems for three-dimensional applications.

An ANC system can be categorized as either feedforward or feedback control. Feedforward ANC systems are classified into

1. broadband control with a reference sensor, which will be discussed in Sect. 49.1, and
2. narrowband control with a reference sensor that is not influenced by the control field, which will be introduced in Sect. 49.2.

The concept of adaptive feedback ANC will be developed in Sect. 49.3 from the standpoint of reference signal synthesis. These single-channel ANC systems will be expanded to multichannel systems in Sect. 49.4.

Traditional acoustic noise control uses passive techniques such as enclosures, barriers, and silencers to attenuate undesired noise. These passive methods achieve high attenuation over a broad frequency range; however, they are relatively large, costly, and ineffective at low frequencies. Active noise control (ANC) [49.1–7] uses an electroacoustic or electromechanical system to cancel the primary (unwanted) noise, based on the principle of superposition. That is, an antinoise of equal amplitude but opposite phase is generated through a secondary source and combined with the primary noise, thus resulting in the cancelation of both noises. ANC is developing rapidly because it efficiently attenuates low-frequency noises, often with potential benefits in size, weight, volume, and cost.

An acoustic ANC system utilizing a microphone and a loudspeaker to generate a canceling sound was first proposed in [49.8]. Since the characteristics of the undesired noise are changing, an ANC system must be adaptive to track these variations. An adaptive filter [49.9, 10] updates its coefficients to minimize an error signal based on a predetermined adaptive algorithm. The most commonly used adaptive filter for practical ANC applications is the finite impulse response (FIR) filter updated by the least-mean-square (LMS) algorithm. An early duct ANC system based on adaptive filter theory was developed in [49.11].

Most ANC systems are digital, where signals from sensors are sampled and processed in real

time using digital signal processing (DSP) systems. Advanced development of DSP chips [49.12] enabled low-cost implementation of powerful adaptive algorithms and thus encouraged real-world application of ANC systems. The continuous progress of ANC involves the development of advanced DSP algorithms, transducers, and hardware. More-sophisticated adaptive algorithms allow faster convergence and greater noise attenuation, and more-powerful DSP hardware allows these advanced ANC systems to be implemented in real time to improve performance.

Many practical considerations arise when an ANC system is deployed in real applications. An approach to analyze ANC performance involves a hierarchy of techniques, starting with an ideal simplified problem and progressively adding practical constraints and other complexities [49.13]. Performance analysis resolves the following issues:

1. fundamental performance limitations
2. practical constraints that limit performance
3. how to balance performance against complexity
4. how to determine a practical design architecture

For industrial applications, the ANC system must have the following properties [49.14]:

1. maximum efficiency over the frequency band to cancel a wide range of noise,
2. the system can be built and preset in the factory and then installed on site,
3. self-adaptability to deal with any variations in the physical parameters, and
4. robustness and reliability.

49.1 Broadband Feedforward Active Noise Control

This section considers broadband feedforward ANC systems that have a single reference sensor, single secondary source, and single error sensor. The general single-channel acoustic ANC system in a duct is illustrated in Fig. 49.1 [49.2], where the reference signal $x(n)$ is picked up by a microphone. This reference signal is processed by the ANC system to generate the secondary signal $y(n)$ to drive a loudspeaker. The error microphone senses the residual noise $e(n)$ and uses it to optimize the performance of the ANC system.

The ANC system shown in Fig. 49.1 can be modeled in an adaptive system identification framework as illustrated in Fig. 49.2 [49.2], in which an adaptive filter $W(z)$ is used to estimate an unknown plant $P(z)$. The primary path $P(z)$ consists of the acoustic response from the reference sensor to the error sensor. The adaptive filter $W(z)$ minimizes the residual error $e(n)$ by generating the filter output $y(n)$ to approximate the primary noise $d(n)$. When $d(n)$ and $y(n)$ are acoustically combined, the residual error is reduced, which results in cancelation of both sounds based on the principle of superposition. The most important difference between Fig. 49.2 and the traditional system identification scheme is the use of an acoustic summing junction instead of the subtraction of $y(n)$ from $d(n)$ in the electrical domain. The performance of ANC is dependent on the coherence of $d(n)$ and $x(n)$. In order to minimize the residual error, it is necessary to maximize the coherence at frequencies for which there is significant noise energy.

The use of an adaptive filter for ANC applications is complicated by the acoustic summing junction, which represents acoustic superposition in the acous-

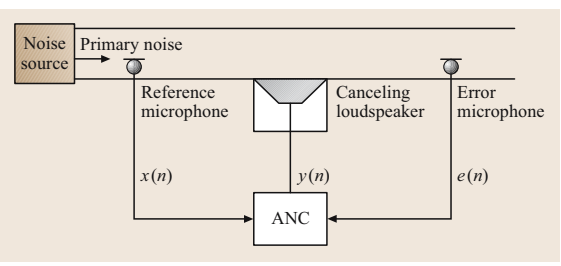


Fig. 49.1 Configuration of single-channel acoustic ANC system in a duct

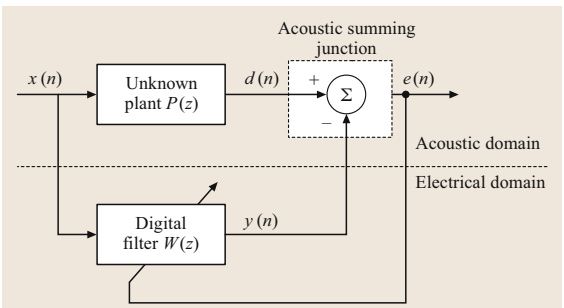


Fig. 49.2 Modeling of ANC using adaptive system identification scheme

tic domain from the canceling loudspeaker to the error microphone. Therefore, it is necessary to compensate for the secondary-path transfer function $S(z)$ from $y(n)$ to $e(n)$, which includes the digital-to-analog converter, reconstruction filter, power amplifier, loudspeaker, the acoustic path from loudspeaker to error microphone, error microphone, preamplifier, anti-aliasing filter, and analog-to-digital converter.

49.1.1 Filtered-X LMS Algorithm

The introduction of the secondary path $S(z)$ into an ANC system using the LMS algorithm will cause instability because the error signal is not correctly *aligned* in time with the reference signal. There are several techniques that can be used to compensate for the effect of $S(z)$, and an effective solution is to place an identical filter in the reference signal path to the weight update of the LMS algorithm, which realizes the filtered-X LMS (FXLMS) algorithm [49.15]. The FXLMS algorithm was independently derived by *Widrow* [49.16] in the context of adaptive control and *Burgess* [49.11] for ANC applications.

The placement of the secondary-path transfer function $S(z)$ following the digital filter $W(z)$ updated by the LMS algorithm is shown in Fig. 49.3 [49.2]. The residual noise is expressed as

$$e(n) = d(n) - s(n) * [\boldsymbol{w}^{\mathrm{T}}(n)\, \boldsymbol{x}(n)] \,, \tag{49.1}$$

where $s(n)$ is the impulse response of secondary path $S(z)$, the asterisk denotes linear convolution, T denotes vector transpose, $\boldsymbol{w}(n) = [w_0(n)\ w_1(n)\ \ldots\ w_{L-1}(n)]^{\mathrm{T}}$ and $\boldsymbol{x}(n) = [x(n)\, x(n-1)\ \ldots\ x(n-L+1)]^{\mathrm{T}}$ are the coefficient and signal vectors, respectively, and L is the filter length.

We assume that the adaptive filter minimizes the instantaneous squared error $e^2(n)$ using the steepest-descent algorithm, which updates the coefficient vector in the negative gradient direction with step size μ. The FXLMS algorithm can then be derived as [49.2]

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) + \mu \boldsymbol{x}'(n) e(n) \,, \tag{49.2}$$

where the step size (or convergence factor) μ determines the convergence speed, $\boldsymbol{x}'(n) = [x'(n)\, x'(n-1)\ \ldots\ x'(n-L+1)]^{\mathrm{T}}$ is the filtered signal vector, and $x'(n) = s(n) * x(n)$. In practical ANC applications, $S(z)$ is unknown and must be estimated by an additional filter $\hat{S}(z)$. Therefore, the filtered reference signal is generated by passing the reference signal $x(n)$ through the secondary-path estimation filter described by

$$\boldsymbol{x}'(n) = \hat{s}(n) * \boldsymbol{x}(n) \,, \tag{49.3}$$

where $\hat{s}(n)$ is the impulse response of the estimated secondary-path filter $\hat{S}(z)$.

Analysis shows that, within the limit of slow adaptation, the algorithm will converge with nearly 90° of phase error between $\hat{S}(z)$ and $S(z)$ [49.15]. Therefore, offline modeling can be used in most practical applications to estimate $S(z)$ during an initial training stage. An experimental setup for offline secondary-path modeling is illustrated in Fig. 49.4 [49.2], where an internally-generated random noise $x(n)$ is used as the input to

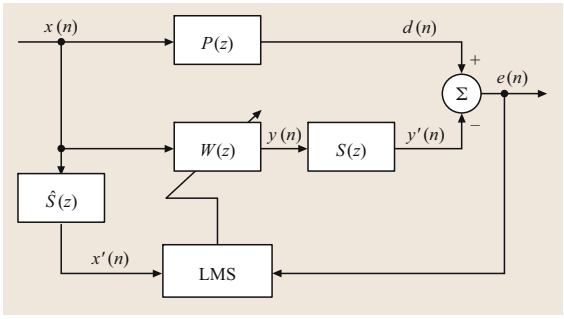


Fig. 49.3 Block diagram of single-channel ANC system using the FXLMS algorithm

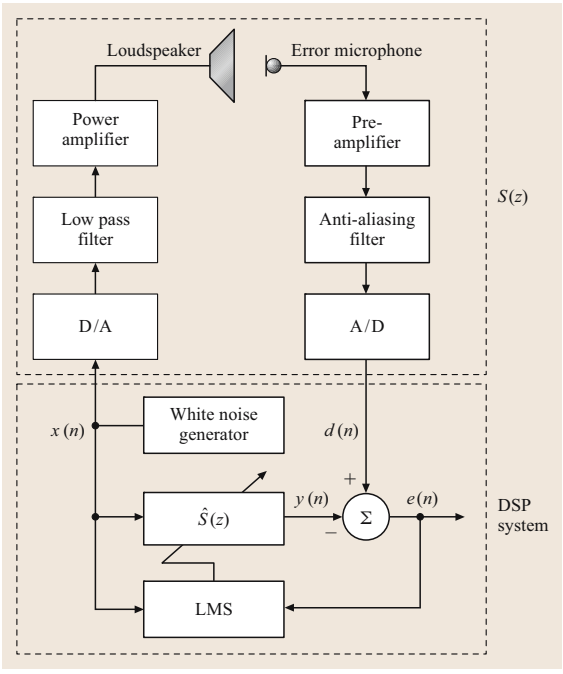


Fig. 49.4 Experimental setup for offline modeling of the secondary path

both the loudspeaker and the adaptive filter $\hat{S}(z)$ updated by the LMS algorithm, which is used to model the secondary path. After convergence of the algorithm, the adaptation is stopped and the coefficients $\hat{s}_l$, $l = 0, 1, \ldots, L-1$ are used to form the secondary-path estimate $\hat{S}(z)$. Algorithms developed for online secondary-path modeling are summarized in [49.2], which may be useful for some applications with significant changes in secondary path during ANC online operation.

49.1.2 Analysis of the FXLMS Algorithm

From Fig. 49.3, the residual error is ideally zero (i. e., $E(z) = 0$) after convergence of the adaptive filter. Therefore, the adaptive filter $W(z)$ will converge to the optimal transfer function

$$W^{\mathrm{o}}(z) = \frac{P(z)}{S(z)} \,. \tag{49.4}$$

This requires the adaptive filter $W(z)$ to simultaneously model $P(z)$ and inversely model $S(z)$. With a correct model, the system can respond instantaneously to changes in the primary noise caused by changes in the noise sources.

Assuming that the adaptive filter $W(z)$ is changing slowly and $\hat{S}(z) = S(z)$, the order of $W(z)$ and $S(z)$ in Fig. 49.3 can be commuted [49.15]. The output of the adaptive filter now carries through directly to the error signal, thus the traditional LMS algorithm analysis method can be used. This method gives accurate results if adaptation is slow by using a small step size μ. The analysis shows that the maximum step size that can be used in the FXLMS algorithm is approximately [49.17]

$$\mu_{\max} = \frac{1}{P_{x'}(L+\Delta)} \,, \tag{49.5}$$

where $P_{x'}$ is the power of the filtered reference signal $x'(n)$, and Δ is the overall delay in the secondary path in terms of the number of samples. Therefore, the secondary-path delay decreases the convergence speed of the ANC system by reducing the maximum step size that can be used in the FXLMS algorithm.

When $\hat{S}(z) \neq S(z)$, the secondary-path estimate has to match the phase within 90° for the algorithm to be stable. The effects of secondary-path modeling errors on the performance of ANC with the FXLMS algorithm are reported in [49.18, 19]. Analysis suggests that phase errors of 40° hardly affect the convergence speed of the algorithm. However, convergence will slow appreciably as the phase difference approaches 90°. Any magnitude estimation error will proportionally change the autocorrelation matrix, and hence scale the ideal stability bound accordingly. However, there is no simple relationship between phase modeling error and stability in the range ±90°.

In a broadband ANC system, measurement noises $u(n)$ and $v(n)$ are present in the reference and error signals, respectively. The optimal unconstrained transfer function $W^{\mathrm{o}}(z)$ becomes [49.2]

$$W^{\mathrm{o}}(z) = \frac{P(z)S_{xx}(z)}{[S_{xx}(z) + S_{uu}(z)]S(z)} \,, \tag{49.6}$$

where $S_{xx}(z)$ and $S_{uu}(z)$ are the power spectra of $x(n)$ and $u(n)$, respectively. This equation shows that $W^{\mathrm{o}}(z)$ is independent of the measurement noise $v(n)$ associated with the error sensor. However, the measurement noise $u(n)$ associated with the reference sensor affects the optimum weight vector, and hence reduces the cancelation performance. The best frequency response of the controller is a compromise between cancelation of the primary noise $x(n)$ and amplification of the measurement noise through the controller [49.14].

If the secondary-path transfer function $S(z)$ is modeled as a pure delay Δ, then $\hat{S}(z)$ in Fig. 49.3 can be replaced by a delay Δ. This special case of the FXLMS algorithm is known as the delayed LMS algorithm [49.20]. The upper bound for the step size depends on the delay Δ and is in close agreement with the approximation given in (49.5). Therefore, efforts should be made to keep the delay small, such as decreasing the distance between the error sensor and the secondary source and reducing the delay in electrical components as shown in Fig. 49.4.

49.1.3 Leaky FXLMS Algorithm

In an ANC application, high noise levels associated with low-frequency resonances may cause nonlinear distortion by overloading the secondary source. A solution to this problem is the introduction of output power constraints. Similar results can be obtained by constraining the adaptive filter weights by modifying the cost function as [49.21]

$$\hat{\xi}(n) = e^2(n) + \gamma \boldsymbol{w}^{\mathrm{T}}(n)\boldsymbol{w}(n) \,, \tag{49.7}$$

where γ is a weighting factor on the control effort. Following the derivation of the FXLMS algorithm, the update algorithm can be derived as [49.2]

$$\boldsymbol{w}(n+1) = \nu \boldsymbol{w}(n) + \mu \boldsymbol{x}'(n)e(n) \,, \tag{49.8}$$

where $\nu = 1 - \mu\gamma$ is the leakage factor. This leaky FXLMS algorithm also reduces numeric error in the finite-precision implementation [49.22]. The introduction of a leakage factor has a considerable stabilizing effect on the adaptive algorithm, especially when very large source strengths are used [49.23].

The leakage has the effect of modifying the autocorrelation matrix of the input process [49.2]. All eigenvalues are positive even if some of the original input eigenvalues are zero. This guarantees a unique solution and a bounded time constant for all modes. The price of leakage is increased complexity of the weight update equation and the introduction of a bias into the converged solution.

49.1.4 Feedback Effects and Solutions

The acoustic ANC system shown in Fig. 49.1 uses a reference microphone to pick up the reference noise. Unfortunately, the antinoise output to the loudspeaker also radiates upstream to the reference microphone, resulting in a corrupted reference signal $x(n)$. The coupling of the acoustic wave from the canceling loudspeaker to the reference microphone is called acoustic feedback.

A more-general block diagram of an ANC system that includes feedback from the secondary source to the reference sensor is shown in Fig. 49.5 [49.2], where $u(n)$ is the primary noise, $x(n)$ is the signal picked up by the reference sensor, and $F(z)$ is the feedback path transfer function from the output of the adaptive filter $W(z)$ to the reference sensor. With the feedback path, the steady-state transfer function of the adaptive filter becomes [49.2]

$$W^{o}(z) = \frac{P(z)}{S(z) + P(z)F(z)} \,. \tag{49.9}$$

The simplest approach to solving the feedback problem is to use a feedback cancelation (or neutralization) filter, which is used for acoustic echo cancelation [49.24]. The electrical model of the feedback path is driven by the secondary signal, and its output estimates the acoustic feedback. A duct acoustic ANC system using the FXLMS algorithm with feedback neutralization is illustrated in Fig. 49.6 [49.2]. The feedback component of the reference signal is canceled electronically using the output of feedback neutralization filter $\hat{F}(z)$, which models the feedback path $F(z)$. Thus, the input signal $x(n)$ is computed as

$$x(n) = u(n) - \sum_{m=0}^{M-1} \hat{f}_m y(n-m-1) \,, \tag{49.10}$$

where $\hat{f}_m$ are the coefficients of the feedback neutralization filter $\hat{F}(z)$ with length M.

Since the primary noise is highly correlated with the antinoise, the adaptation of the feedback neutralization filter must be inhibited when the ANC system is in operation. This is similar to adaptive echo cancelation during periods of double-talk. Thus, the feedback neutralization filter can be obtained by using an offline modeling method for estimating the transfer function of the feedback path. Moreover, the models $\hat{S}(z)$ and $\hat{F}(z)$ can be estimated simultaneously using offline modeling [49.2].

Equation (49.9) shows that the optimal solution of the adaptive filter is generally an infinite impulse response (IIR) function. The poles of an IIR filter provide a well-matched solution with a lower-order filter. However, the disadvantages of adaptive IIR filters are

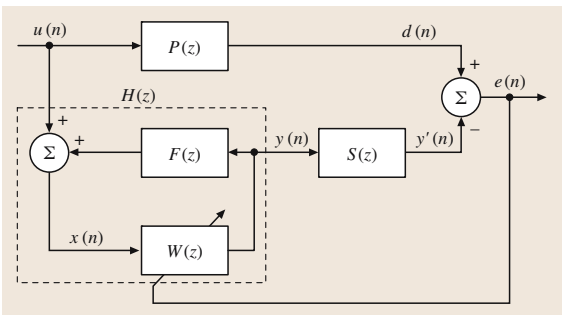


Fig. 49.5 Block diagram of ANC with acoustic feedback

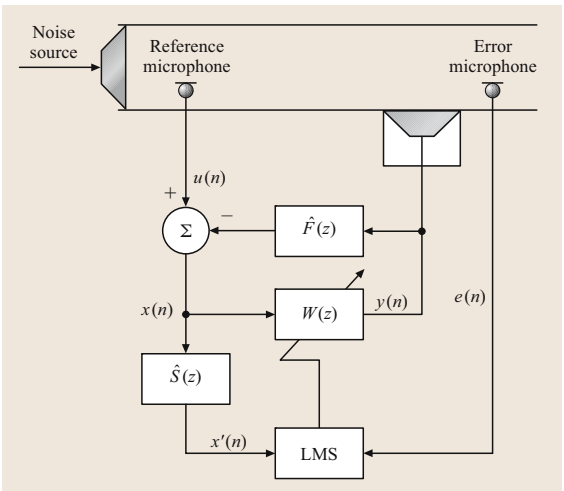


Fig. 49.6 ANC system with acoustic feedback cancelation

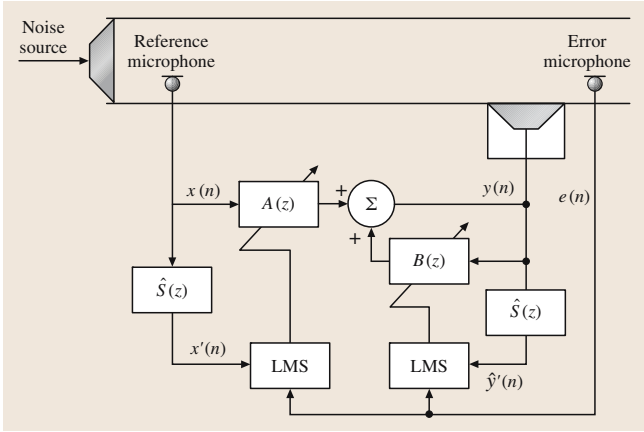


Fig. 49.7 ANC system using an adaptive IIR filter

1. IIR filters may be unstable if some pole(s) of the filter move outside of the unit circle during the adaptive process,
2. the adaptation may converge to a local minimum because the cost function of adaptive IIR filters is generally nonquadratic, and
3. IIR adaptive algorithms can have a relatively slow convergence rate in comparison with that of FIR filters.

A comprehensive discussion of adaptive IIR filters can be found in [49.25].

A block diagram of an ANC system using an adaptive IIR filter [49.26] is illustrated in Fig. 49.7 [49.2]. The output signal of the IIR filter $y(n)$ is computed as

$$y(n) = \boldsymbol{a}^{\mathrm{T}}(n)\boldsymbol{x}(n) + \boldsymbol{b}^{\mathrm{T}}(n)\boldsymbol{y}(n-1)\,, \tag{49.11}$$

where $\boldsymbol{a}(n) \equiv [a_0(n)\, a_1(n)\, \ldots\, a_{L-1}(n)]^{\mathrm{T}}$ is the weight vector of $A(z)$, $\boldsymbol{x}(n)$ is the reference signal vector, $\boldsymbol{b}(n) \equiv [b_1(n)\, b_2(n)\, \ldots\, b_M(n)]^{\mathrm{T}}$ is the weight vector of $B(z)$, and $\boldsymbol{y}(n-1)$ is the output signal vector delayed by one sample. Based on the recursive LMS algorithm [49.27], the filtered-U recursive LMS algorithm for ANC is derived as [49.28]

$$\boldsymbol{a}(n+1) = \boldsymbol{a}(n) + \mu \boldsymbol{x}'(n) e(n)\,, \tag{49.12}$$

$$\boldsymbol{b}(n+1) = \boldsymbol{b}(n) + \mu \hat{\boldsymbol{y}}'(n-1) e(n)\,, \tag{49.13}$$

where $\hat{\boldsymbol{y}}'(n-1) = \hat{s}(n) * \boldsymbol{y}(n-1)$ is the filtered version of the canceling signal vector at time $n-1$. Real-time experiments have been conducted to test the system performance for various reference microphone locations, error microphone locations, and different time-varying sources, such as a centrifugal fan and diesel engine [49.29].

49.2 Narrowband Feedforward Active Noise Control

Many noises generated by engines, compressors, motors, fans, and propellers are periodic. Direct observation of the rotation of such sources is generally possible by using an appropriate sensor, which provides information for generating an electrical reference signal that contains the fundamental frequency and all the harmonics of the primary noise.

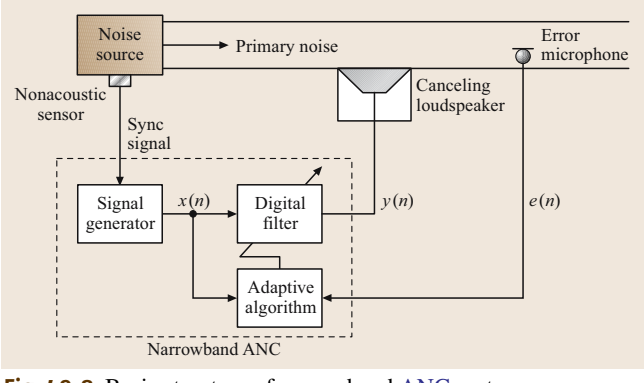


Fig. 49.8 Basic structure of narrowband ANC system

49.2.1 Introduction

A basic block diagram of narrowband ANC for reducing periodic acoustic noise in a duct is illustrated in Fig. 49.8. This system uses a synthesized reference signal $x(n)$ internally generated by the ANC system. This technique has the following advantages:

1. undesired acoustic feedback from the canceling loudspeaker back to the reference microphone is avoided,
2. nonlinearities and aging problems associated with the reference microphone are avoided,
3. the periodicity of the noise removes the causality constraint,

4. the use of an internally generated reference signal results in the ability to control each harmonic independently, and
5. it is only necessary to model the transfer function over frequencies of the harmonic tones; thus, an FIR filter with substantially lower order may be used.

As shown in Fig. 49.8 [49.2], the reference signal generator is triggered by a synchronization signal from a nonacoustic sensor, such as a tachometer from an automotive engine. Two types of reference signals are commonly used in narrowband ANC systems:

1. an impulse train with a period equal to the inverse of the fundamental frequency of the periodic noise [49.30]
2. sine waves that have the same frequencies as the corresponding tones to be canceled

The first technique is called the waveform synthesis method [49.31], and the second technique is called the adaptive notch filter, which was originally developed for the cancelation of tonal interference [49.32].

49.2.2 Waveform Synthesis Method

The waveform synthesizer stores canceling noise waveform samples $\{w_l(n), l = 0, 1, \ldots, L-1\}$ in memory, where L is the number of samples over one cycle of the waveform. These samples represent the required waveform, and are sequentially sent to the secondary loudspeaker producing the actual canceling noise waveform. The data pointer to the memory buffer address can be incremented in a circular fashion between 0 and $L-1$ for each sampling period T, controlled by interrupts generated from the synchronization signal. It is important to note that most advanced digital signal processors support circular addressing in hardware [49.12].

The residual noise picked up by the error microphone is also synchronously sampled with the reference signal. In a practical system, there is a delay between the time the signal $y(n)$ is fed to the loudspeaker and the time it is received at the error microphone. This delay can be accommodated by subtracting a time offset from the circular pointer. The waveform synthesis method is equivalent to an adaptive FIR filter of order $L = N$ excited by a Kronecker impulse train of period $N = T_0/T$ samples [49.30], where $T_0 = 2\pi/\omega_0$ is the period of the noise with fundamental frequency ω_0.

For an adaptive filter with length $L = N$, the transfer function $H(z)$ between the primary input $D(z)$ and the error output $E(z)$ is derived as [49.30]

$$H(z) = \frac{E(z)}{D(z)} = \frac{1 - z^{-L}}{1 - (1-\mu)z^{-L}} \,. \tag{49.14}$$

The zeros have constant amplitude ($|z| = 1$) and are equally spaced ($2\pi/L$) on the unit circle of the z-plane to create nulls at frequencies $k\omega_0$. Therefore, the tonal components of the periodic noise at the fundamental and harmonic frequencies are attenuated by this multiple notch filter. The poles have the same frequencies as the zeros, but are equally spaced on a circle at distance $(1-\mu)$ from the origin. The effect of the poles is to reduce the bandwidth of the notch.

Equation (49.14) suggests the use of $0 < \mu < 1$ to guarantee stability. The 3 dB bandwidth of each notch for $\mu \ll 1$ is approximated as $B \approx \mu/\pi T$ (Hz) [49.30]. Thus, the bandwidth of the notch filter is proportional to the step size μ. However, the time constant of the response envelope decay is approximately $\tau \approx T/\mu$ (s). Therefore, there is a tradeoff between the notch bandwidth and the duration of the transient response, which is determined by the step size and the sampling rate of the narrowband ANC system.

As discussed in Sect. 49.1, the effects of the secondary path $S(z)$ must be compensated for by using the FXLMS algorithm. The presence of $S(z)$ modifies the transfer function of the controller to [49.33]

$$H(z) = \frac{1 - z^{-L}}{1 - [1 - \mu S(z)]z^{-L}} \,. \tag{49.15}$$

For the case of a single sinusoid, the secondary path can be modeled by a pure delay. Therefore, the compensator can be approximated as $\hat{S}(z) = z^{-\Delta}$, where Δ is the number of samples of delay from $y(n)$ to $e(n)$. The behavior of the synchronous periodic ANC with delayed coefficient update changes significantly as the delay exceeds each integer multiple of L. The steady-state transfer function $H(z)$ from $D(z)$ to $E(z)$ for the delayed LMS algorithm is [49.34]

$$H(z) = \frac{1 - z^{-L}}{1 - (1 - \mu z^{-\lfloor \Delta/L \rfloor L})z^{-L}} \,. \tag{49.16}$$

This transfer function shows that the delayed coefficient adaptation changes the pole structure. In addition to reducing the bandwidth of the notch, the inclusion of delay shifts the angle of the poles away from $k\omega_0$, and accordingly increases the out-of-band overshoot of the frequency response [49.35]. As the step size μ and/or the delay in the secondary path are increased, these out-of-band peaks become larger until the system finally becomes unstable.

49.2.3 Adaptive Notch Filters

An adaptive notch filter can be realized by using an adaptive filter with a sinusoidal reference signal. The advantages of the adaptive notch filter are that it offers easy control of bandwidth, an infinite null, and the capability to track the exact frequency of the interference adaptively. The reference input is a cosine wave $x(n) = x_0(n) = A\cos(\omega_0 n)$, where A and ω_0 are the amplitude and frequency, respectively, of the reference signal. A 90° phase shifter is used to produce the quadrature reference signal $x_1(n) = A\sin(\omega_0 n)$. The steady-state transfer function $H(z)$ from the primary input $d(n)$ to the noise canceler output $e(n)$ is [49.32]

$$H(z) = \frac{E(z)}{D(z)} = \frac{z^2 - 2z\cos\omega_0 + 1}{z^2 - (2-\mu A^2)z\cos\omega_0 + 1 - \mu A^2} \,. \quad (49.17)$$

The zeros of $H(z)$ are located on the unit circle in the z-plane at $z = \mathrm{e}^{\pm \mathrm{i}\omega_0}$. The adaptive noise canceler therefore acts as a tunable notch filter, with a notch located at the reference frequency ω_0. For a general adaptive filter of length L, (49.17) becomes [49.36]

$$H(z) = \frac{z^2 - 2z\cos\omega_0 + 1}{z^2 - \left(2 - \frac{\mu L A^2}{2}\right) z\cos\omega_0 + 1 - \frac{\mu L A^2}{2}} \,. \quad (49.18)$$

The sharpness of the notch is determined by the closeness of the poles to the zeros. The 3 dB bandwidth of the notch filter is estimated as [49.36]

$$B \approx \frac{\mu L A^2}{4\pi T} \quad \text{(Hz)} \,. \quad (49.19)$$

When the interfering sinusoid frequency changes rapidly or jitters around, the notch width must cover a wider frequency range. This can be accomplished by using a larger μ, which has the effect of providing faster tracking.

For $L = 2$, the autocorrelation matrix $\boldsymbol{R}$ is a diagonal matrix where the diagonal terms are equal to $A^2/2$ and the off-diagonal terms are zero [49.2]. The eigenvalues λ_1 and λ_2 of the $\boldsymbol{R}$ matrix are identical. Therefore, the system has very fast convergence since the eigenvalue spread equals 1. The $1/\mathrm{e}$ time constant of the adaptation is approximated as

$$\tau_{\mathrm{mse}} \le \frac{T}{\mu\lambda} = \frac{2T}{\mu A^2} \quad \text{(s)} \,, \quad (49.20)$$

which is determined by the amplitude square of the reference sine wave and the step size μ.

The application of the adaptive notch filter to periodic ANC has been developed by *Ziegler* [49.37]. A recursive quadratic oscillator provides two orthogonal components $x_0(n)$ and $x_1(n)$, which are used as reference inputs for the adaptive filter. These two signals are separately weighted and then summed to produce the canceling signal $y(n)$. The delayed LMS algorithm updates the filter weights to minimize the residual error $e(n)$:

$$w_l(n+1) = w_l(n) + \mu e(n) x_l(n-\Delta) \,, \quad l = 0, 1 \,, \quad (49.21)$$

where Δ is used to compensate for the secondary path. This frequency-dependent delay can be determined offline from the estimated secondary path.

The delay unit can be replaced by a secondary-path estimate $\hat{S}(z)$ as in the FXLMS algorithm illustrated in Fig. 49.9 [49.2]. The adaptive weights are updated as

$$w_l(n+1) = w_l(n) + \mu x_l'(n) e(n), \quad l = 0, 1 \,, \quad (49.22)$$

where $x_0'(n)$ and $x_1'(n)$ are the filtered versions of $x_0(n)$ and $x_1(n)$, respectively. In the limit of slow adaptation, the transfer function of the narrowband ANC system then becomes [49.2]

$$H(z) = \frac{z^2 - 2z\cos\omega_0 + 1}{z^2 - [2\cos\omega_0 - \beta\cos(\omega_0 - \phi_\Delta)]z + 1 - \beta\cos\phi_\Delta} \,, \quad (49.23)$$

where $\beta = \mu A^2 A_s$, A_s is the amplitude of $S(z)$ at frequency ω_0, and $\phi_\Delta = \phi_s - \phi_{\hat{s}}$ is the phase difference between $S(z)$ and $\hat{S}(z)$ at ω_0. For small β, $H(z)$ has complex conjugate poles at radius $r_\mathrm{p} = \sqrt{1 - \beta\cos\phi_\Delta}$. Since all the terms composing β are positive, the radius of the pole can be greater than 1 only if $\cos\phi_\Delta$ is negative. Accordingly, the stability condition is

$$-90° < \phi_\Delta < 90° \quad (49.24)$$

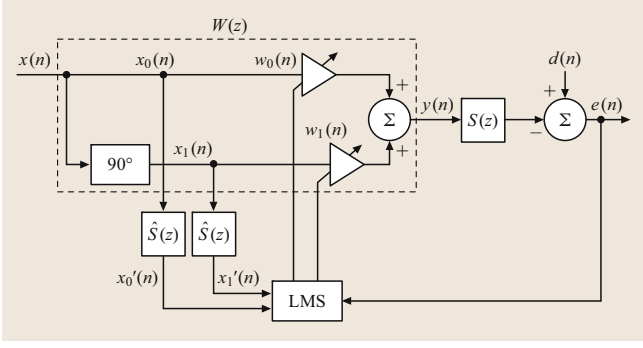


Fig. 49.9 Single-frequency ANC system using the FXLMS algorithm

and the convergence time constant is slowed down by a factor of $1/\cos\phi_\Delta$ [49.15].

Another method for analyzing the stability and transient response of the adaptive notch filter using the FXLMS algorithm is to formulate the problem in the complex weight domain and apply standard control theory [49.38]. Application of this analysis technique showed that large out-of-band gain can lead to instability. The origin and characteristics of asymmetric out-of-band overshoot were investigated in [49.39] using a perturbation technique to quantify the pole displacements due the secondary path. One solution to the out-of-band gain problem is to equalize the secondary-path transfer function $S(z)$ in phase and amplitude over the entire band. An alternative solution is to control the out-of-band response by either employing a band-pass filter in the secondary path before demodulation or a low-pass filter after demodulation [49.38]. However, there is an inherent tradeoff here because in addition to attenuating out-of-band gain, a band-pass filter will also introduce delay. Another solution consists of two interconnected adaptive filters using the same internally generated reference signal [49.40].

49.2.4 Multiple-Frequency ANC

In practical ANC applications, periodic noise usually contains multiple tones at several harmonic-related frequencies. This type of noise can be attenuated by a filter with multiple notches. An adaptive filter with multiple notches can be implemented by direct, parallel, direct/parallel, or cascade form.

A method for eliminating multiple sinusoidals or other periodic interference was proposed by *Glover* [49.36]. The reference signal is a sum of M sinusoids:

$$x(n) = \sum_{m=1}^{M} A_m \cos(\omega_m n) \,, \tag{49.25}$$

where A_m and ω_m are, respectively, the amplitude and the frequency of the m-th sinusoid. When the frequencies of the reference sinusoids are close together, a long filter ($L \gg 2M$) is required to give good resolution between adjacent frequencies. This is an undesired solution since a higher-order adaptive FIR filter results in slower convergence, higher excess mean-square error, and larger numeric errors. An application of Glover's method for actively attenuating engine-generated noise was proposed in [49.41]. The relationship between the convergence rate and required filter length with the frequency separation between two adjacent harmonics is studied in [49.42] based on the analysis of eigenvalue spread.

If the primary noise contains M sinusoids, M two-weight adaptive filters can be connected in parallel to attenuate these narrowband components [49.37]. The canceling signal is a sum of M adaptive filter outputs $y_m(n)$:

$$y(n) = \sum_{m=1}^{M} y_m(n) \,. \tag{49.26}$$

Since only one error sensor is used, there is only one error signal $e(n)$ to update all M adaptive filters based on the FXLMS algorithm.

A configuration using multiple reference signal generators and corresponding adaptive filters has been developed [49.43] to improve the performance of ANC systems. The idea is to separate a collection of many harmonically related sinusoids into mutually exclusive sets that individually have frequencies spaced out as far as possible. In general, if there are M harmonics to be canceled and K ($K < M$) signal generators are used, each reference signal $x_k(n)$, $k = 1, 2, \ldots, K$ contains staggered sinusoidal frequencies of every other K-th harmonic. These reference signals are processed by their corresponding adaptive filters. By partitioning signal component frequencies in this fashion, the rate of convergence of each adaptive filter can be significantly improved. This is because the frequency difference between any two adjacent sinusoidal components in $x_k(n)$ is effectively increased, as compared to the direct implementation technique [49.42].

Ideally, multiple-sinusoid references are more effectively employed in a cascade of M second-order single-frequency notch filters. The overall response of such an arrangement is given by [49.35]

$$H(z) = \prod_{m=1}^{M} H_m(z) = \prod_{m=1}^{M} \frac{1}{1 + S(z)W_m(z)} \,, \tag{49.27}$$

where $W_m(z)$ is the m-th section adaptive filter. Each $W_m(z)$ produces a pole at frequency ω_m, and so each $H_m(z)$ produces a notch at ω_m. If an estimate of the secondary-path transfer function is available, it is possible to configure a *pseudocascade* arrangement [49.35] that ideally performs as a true cascade but requires only one secondary-path estimate $\hat{S}(z)$.

A rectangular wave $x(t)$ potentially contains the fundamental and all harmonic components of the periodic noise. The shape of the spectrum of $x(t)$ is dependent on the duty-cycle ratio τ/T_0, where τ

and T_0 are the pulse width and the fundamental period of the rectangular wave. When τ approaches zero, the rectangular wave becomes an impulse train. Usually satisfactory performance can be achieved for [49.2]

$$\frac{\tau}{T_0} \leq \frac{1}{2M} , \tag{49.28}$$

where M is the largest significant harmonic index of the primary noise. Reference-signal components above the largest significant harmonic must be removed by a low-pass filter in order to avoid aliasing problems. Applications of this principle were developed for attenuating engine noise in an earth-moving machine [49.44] and tonal blade passage noise in a multiple-blade fan [49.45].

49.2.5 Active Noise Equalization

The design of an ANC system usually pursues maximal attenuation of the primary noise. However, in some applications, it is desirable to retain a small residual error with a specified spectral shape. An ANC system that is capable of controlling residual noise spectrum shape is called an active noise equalizer [49.46]. The principle of narrowband active noise equalization can also be applied to broadband noise [49.47].

In a narrowband active noise equalizer, the output of the two-weight filter, $y(n)$, is split into two branches. The gains β and $(1-\beta)$ are inserted in these two branches to adjust the residual noise. If the secondary-path estimate is perfect (i. e., $\hat{S}(z) = S(z)$), the pseudo-error $e'(n) = d(n) - y(n)$ is the residual noise of the conventional ANC system. The adaptive filter minimizes the pseudo-error using the FXLMS algorithm. After the filter has converged, the pseudo-error $e'(n) \approx 0$, however, the real residual noise is $e(n) \approx \beta d(n)$. Thus, $e(n)$ contains a residual narrowband noise whose amplitude is totally controlled by adjusting the gain value β.

The functions of the active noise equalizer can be classified into the following four operation modes:

1. Cancelation mode: $\beta = 0$, the active noise equalizer attenuates the sinusoid completely at frequency ω_0;
2. Attenuation mode: $0 < \beta < 1$, the amount of attenuation is determined by the factor β;
3. Neutral mode: $\beta = 1$, the noise was passed without attenuation;
4. Enhancement mode: $\beta > 1$, the active noise equalizer functions as an amplifier.

49.3 Feedback Active Noise Control

In a single-channel feedback ANC system, the error sensor output is processed by an ANC system to generate the secondary signal [49.1]. A single-channel adaptive feedback ANC system was proposed in [49.48] and extended to the multichannel case in [49.49]. We view this technique as an adaptive feedforward system that synthesizes (or regenerates) its own reference signal based only on the adaptive filter output and error signal. Adaptive feedback ANC has been applied in the design of active headsets [49.50, 51].

In Fig. 49.3, the primary noise is expressed in the z-domain as $D(z) = E(z) + S(z)Y(z)$, where $E(z)$ is the signal obtained from the error sensor and $Y(z)$ is the secondary signal generated by the adaptive filter. If $\hat{S}(z) \approx S(z)$, we can estimate the primary noise $d(n)$ and use this as a synthesized reference signal $x(n)$. That is,

$$X(z) \equiv \hat{D}(z) = E(z) + \hat{S}(z)Y(z) . \tag{49.29}$$

This is the principle for reference signal synthesis (regeneration) technique, whereby the secondary signal $y(n)$ is filtered by the secondary-path estimate $\hat{S}(z)$ and then combined with $e(n)$ to regenerate the primary noise.

The complete single-channel adaptive feedback ANC system using the FXLMS algorithm is illustrated in Fig. 49.10 [49.2], where $\hat{S}(z)$ is also required to com-

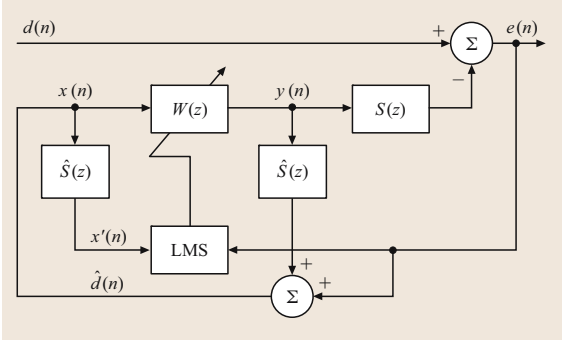


Fig. 49.10 Adaptive feedback ANC system using the FXLMS algorithm

pensate for the secondary path. The reference signal $x(n)$ is synthesized as

$$x(n) \equiv \hat{d}(n) = e(n) + \sum_{m=0}^{M-1} \hat{s}_m y(n-m) \,, \qquad (49.30)$$

where $\hat{s}_m$, $m = 0, 1, \ldots, M-1$ are the coefficients of the FIR filter $\hat{S}(z)$ used to estimate the secondary path.

From Fig. 49.10, we have $x(n) = d(n)$ if $\hat{S}(z) = S(z)$. If the step size μ of the LMS algorithm is small (slow convergence), the adaptive filter $W(z)$ can be commuted with $S(z)$ [49.9]. If we further assume that the secondary path $S(z)$ can be modeled by a pure delay, that is, $S(z) = z^{-\Delta}$, Fig. 49.10 is identical to the standard adaptive prediction scheme. The system response from $d(n)$ to $e(n)$ is called the prediction error filter $H(z)$. The adaptive filter $W(z)$ acts as an adaptive predictor of the primary noise $d(n)$ to minimize the residual noise $e(n)$, so the performance of the adaptive feedback ANC system depends on the predictability of the primary noise $d(n)$.

The feedforward ANC systems discussed in Sect. 49.1 use a reference sensor to measure the primary noise and an error sensor to monitor the performance. The adaptive feedback ANC system only uses an error sensor and thus cancels only the predictable noise components of the primary noise. A combination of the feedforward and feedback control structures is called a hybrid ANC system with both reference and error sensors, as illustrated in Fig. 49.11 [49.52]. The reference sensor is kept close to the noise source and provides a coherent reference signal for the feedforward ANC system. The error sensor is placed downstream

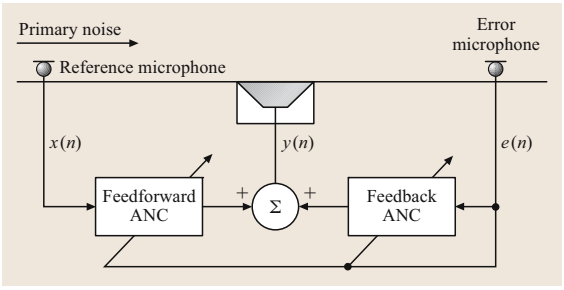


Fig. 49.11 Combination of feedforward and adaptive feedback ANC with the FXLMS algorithm

and senses the residual noise, which is used to synthesize the reference signal for the adaptive feedback ANC filter, as well as to adapt the coefficients of both the feedforward and feedback ANC filters. The feedforward ANC attenuates primary noise that is correlated with the reference signal, while the feedback ANC cancels the predictable components of the primary noise that are not observed by the reference sensor.

In the hybrid ANC system, the secondary signal $y(n)$ is generated using the outputs of both the feedforward and feedback ANC filters. A similar hybrid ANC system using the adaptive IIR feedforward ANC and the adaptive feedback ANC can be found in [49.2]. The advantage of hybrid ANC over other ANC systems is that a lower-order filter can be used to achieve the same performance. The hybrid systems also clearly demonstrate an advantage over either the simple feedforward ANC or adaptive feedback ANC system alone when there is significant plant noise.

49.4 Multichannel ANC

The noise field in an enclosure or a large-dimension duct is more complicated than in a narrow duct, so it is generally necessary to use a multichannel ANC system with several secondary sources, error sensors, and perhaps even several reference sensors.

49.4.1 Principles

Theoretical predictions, computer simulations, and laboratory experiments on harmonic ANC in a shallow enclosure were presented in [49.53–55]. The total acoustic potential energy E_p is approximated by the sum of the squares of the outputs of many error sensors distributed throughout the enclosure. This suggests that the multichannel ANC system should minimize a cost function

$$\xi = \frac{V}{4\rho c^2 M} \sum_{m=1}^{M} |p_m|^2 \,, \qquad (49.31)$$

where V is the volume, ρ is the density of the acoustic medium, c is the propagation velocity, and p_m is the sound pressure at the m-th error sensor with a total of M error microphones. Equation (49.31) shows that ξ is a quadratic function, and thus allows the efficient gradient descent method to be employed in the multichannel ANC algorithm.

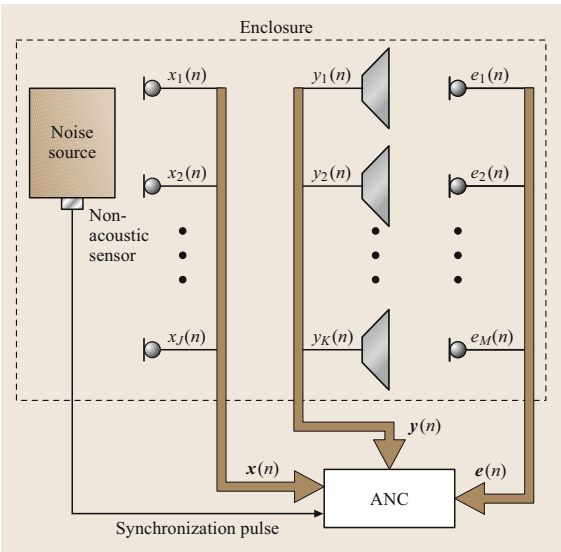


Fig. 49.12 Multichannel acoustic ANC system

For canceling low-frequency tones, only a few error sensors are needed to achieve a substantial reduction in total noise energy [49.56]. The locations of these error sensors are very important in order to obtain the best estimate of the total acoustic potential energy. The secondary sources should be located in positions where they can couple well to the acoustic modes of the enclosure. To improve the performance of the system, the distance from the primary to the secondary sources must be less than a quarter-wavelength at the highest frequency [49.57]. Subtle physical attributes of the multichannel ANC system that affect convergence are the placement of the error sensors and the location of the secondary sources.

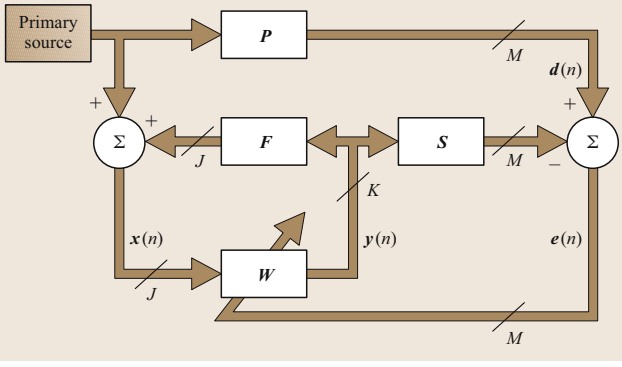


Fig. 49.13 Multichannel FXLMS algorithm

49.4.2 Multichannel FXLMS Algorithms

A multichannel acoustic ANC system is illustrated in Fig. 49.12 [49.2]. For narrowband feedforward multichannel ANC systems, a nonacoustic sensor can be used as the reference sensor to generate a reference signal. For broadband feedforward ANC, the multichannel system employs J reference sensors to form the reference signal vector. For adaptive feedback ANC, the reference signals are internally synthesized based on the secondary and error signals. The multichannel ANC system generates K canceling signals to drive the corresponding secondary sources, and M error sensors are distributed over the desired locations to measure the residual noise components.

A block diagram of a multichannel ANC system that includes feedback paths from the secondary sources to the reference sensors is shown in Fig. 49.13 [49.2]. The wide arrows represent an array of signals (acoustic or electrical) that are symbolically expressed as vectors. The matrix $\boldsymbol{P}$ represents $M \times J$ primary-path transfer functions from the primary source to each error sensor output $e_m(n)$. The matrix $\boldsymbol{S}$ represents $M \times K$ secondary-path transfer functions, $S_{mk}(z)$, from K secondary sources to M error sensors. Also, the matrix $\boldsymbol{F}$ represents $J \times K$ feedback paths from K secondary sources to J reference sensors. There are $K \times J$ possible feedforward channels, each demanding a separate adaptive filter, and these $K \times J$ adaptive filters are represented by the matrix $\boldsymbol{W}$. A complete set of multichannel FXLMS algorithms for broadband feedforward, narrowband feedforward, and adaptive feedback ANC systems are introduced in [49.2].

The single-reference/multiple-output ANC system aims to control a multidimensional noise field produced by rotating machinery such as engines [49.21]. The k-th secondary signal $y_k(n)$ is obtained by filtering the reference signal $x(n)$ through the corresponding adaptive FIR filter:

$$y_k(n) = \boldsymbol{w}_k^{\mathrm{T}}(n)\boldsymbol{x}(n)\,, \quad k = 1, 2, \ldots, K\,, \qquad (49.32)$$

where $\boldsymbol{w}_k(n) \equiv [w_{k,0}(n)\ w_{k,1}(n)\ \ldots\ w_{k,L-1}(n)]^{\mathrm{T}}$ is the weight vector of the k-th adaptive filter and $\boldsymbol{x}(n) = [x(n)\, x(n-1)\ \ldots\ x(n-L+1)]^{\mathrm{T}}$ is the common reference signal vector for all adaptive filters.

The single-reference/multiple-output FXLMS algorithm can be partitioned into K equations [49.2]

$$\boldsymbol{w}_k(n+1) = \boldsymbol{w}_k(n) + \mu \sum_{m=1}^{M} \boldsymbol{x}'_{km}(n) e_m(n)\,,$$
$$k = 1, 2, \ldots, K\,, \qquad (49.33)$$

where

$$\boldsymbol{x}'_{km}(n) \equiv \hat{s}_{mk}(n) * \boldsymbol{x}(n) \,, \tag{49.34}$$

for $k = 1, 2, \ldots, K$ and $m = 1, 2, \ldots, M$, are the filtered reference signal vectors, which are formed by filtering $\boldsymbol{x}(n)$ by the secondary-path estimates $\hat{S}_{mk}(z)$ from the k-th secondary source to the m-th error sensor. In (49.34), $\hat{s}_{mk}(n)$ represents the impulse response of an FIR filter $\hat{S}_{mk}(z)$ that is used to estimate $S_{mk}(z)$. The multichannel leaky FXLMS algorithm is similar to (49.33), except that $\nu \boldsymbol{w}_k(n)$ is used to replace $\boldsymbol{w}_k(n)$.

In the general multiple-reference/multiple-output ANC system using the FXLMS algorithm, the ANC filter $\boldsymbol{W}$ has J reference input signals $x_j(n)$. Each controller in the matrix $\boldsymbol{W}$ is represented by $W_{kj}(z)$, where j is the reference input index and k is the secondary source index. The secondary signal output to the k-th secondary source is

$$y_k(n) = \sum_{j=1}^{J} \boldsymbol{w}_{kj}^{\mathrm{T}}(n) \boldsymbol{x}_j(n) \,,$$
$$k = 1, 2, \ldots, K \,, \tag{49.35}$$

where $\boldsymbol{x}_j(n) \equiv [x_j(n)\, x_j(n-1) \ldots x_j(n-L+1)]^{\mathrm{T}}$, $j = 1, 2, \ldots, J$ are the reference signal vectors. There are $M \times K$ different secondary paths $S_{mk}(z)$ between the secondary sources and error sensors, which are modeled by $\hat{S}_{mk}(z)$ to generate an array of filtered reference signals $\boldsymbol{x}'_{jkm}(n)$ for the multichannel FXLMS algorithm

$$\boldsymbol{w}_{kj}(n+1) = \boldsymbol{w}_{kj}(n) + \mu \sum_{m=1}^{M} \boldsymbol{x}'_{jkm}(n) e_m(n) \tag{49.36}$$

for $k = 1, 2, \ldots, K$ and $j = 1, 2, \ldots, J$, where the vectors

$$\boldsymbol{x}'_{jkm}(n) = \hat{s}_{mk}(n) * \boldsymbol{x}_j(n) \tag{49.37}$$

are the filtered reference signal vectors.

A practical application of multiple-reference/multiple-output active control for propeller blade-passage noise inside a 50 seat aircraft has been reported [49.17]. That system uses three reference signals (internally generated sinusoids) for the fundamental frequency and its first two harmonics, 16 secondary sources, and 32 error sensors, resulting in a $3 \times 16 \times 32$ multiple-reference/multiple-output ANC system. Since the reference signal is an internally generated sinusoid, only two coefficients are required for each of the 1536 ($3 \times 16 \times 32$) individual secondary-path models to generate the filtered reference signals.

In some cases, it is advantageous to locally minimize the sum of the squared signals from only a few error sensors. This concept is formalized by employing independent weighting factors q_{km} to control the influence of the M error signals on each of the K adaptive filters [49.58]. The single-reference/multiple-output FXLMS algorithm given in (49.33) is then modified by using $e'_{km}(n)$ to replace $e_{km}(n)$, where

$$e'_{km}(n) \equiv q_{km} e_m(n) \tag{49.38}$$

is the error signal $e_m(n)$ weighted by q_{km} for the adaptation of $\boldsymbol{w}_k(n)$. Thus, the influence of the error signals is limited by setting some of the q_{km} equal to zero.

49.4.3 Frequency-Domain Convergence Analysis

The convergence behavior of a multichannel ANC system can be analyzed in the frequency domain in terms of the convergence of the individual secondary signals, the cost function, and the control effort [49.56]. This frequency-domain analysis provides insight into the physical process of controlling one or more harmonics in narrowband ANC systems.

To cancel a narrowband noise component of known frequency, the reference signal is a pure sinusoid, which can be considered a constant of value $X = 1$ in the frequency domain. The multichannel frequency-domain FXLMS algorithm is expressed as [49.56]

$$\boldsymbol{W}(n+1) = (\boldsymbol{I} - \mu \boldsymbol{S}^{\mathrm{H}} \boldsymbol{S}) \boldsymbol{W}(n) + \mu \boldsymbol{S}^{\mathrm{H}} \boldsymbol{D} \,, \tag{49.39}$$

where the superscript H (Hermitian) denotes conjugate transpose. Therefore, the eigenvalues of $\boldsymbol{S}^{\mathrm{H}} \boldsymbol{S}$ determine the convergence characteristics of the multichannel frequency-domain FXLMS algorithm. It is noteworthy that disparate eigenvalues arise here because of the spatial nature of the secondary-path matrix $\boldsymbol{S}$. Thus, the convergence of the general broadband multichannel FXLMS algorithm is limited by both the spatial and temporal characteristics of the system.

As with the single-channel leaky FXLMS algorithm, the cost function can be modified to include the control effort as

$$\xi(n) = \boldsymbol{E}^{\mathrm{H}} \boldsymbol{E} + \gamma \boldsymbol{W}^{\mathrm{H}} \boldsymbol{W} \,, \tag{49.40}$$

where γ is a real constant that determines the balance between reducing the total mean-square error $\boldsymbol{E}^{\mathrm{H}} \boldsymbol{E}$ and moderating the control effort $\boldsymbol{W}^{\mathrm{H}} \boldsymbol{W}$. The convergence

of the multichannel frequency-domain leaky FXLMS algorithm is guaranteed if [49.56]

$$\mu < \frac{2}{\sigma_k^2 + \gamma}\,, \quad k = 1, 2, \ldots, K\,, \tag{49.41}$$

where σ_k^2 is the k-th eigenvalue of $\boldsymbol{S}^{\mathrm{H}}\boldsymbol{S}$. The effect of γ on the convergence of the ANC system is to speed up the very slow modes of convergence, which are associated with small values of σ_k^2, by adding a constant factor γ.

Consider the unconstrained solution for $\gamma = 0$. In this case the effort can be written [49.56]

$$\boldsymbol{W}^{\mathrm{H}}(n)\boldsymbol{W}(n) \approx \sum_{k=1}^{K} \frac{|P_k|^2}{\sigma_k^2}\left[1 - \left(1 - \mu\sigma_k^2\right)^n\right]^2\,. \tag{49.42}$$

For $\mu\sigma_k^2 \ll 1$, the m-th mode requires a control effort of $|P_k|^2/\sigma_k^2$ and it converges exponentially with a time constant $1/\mu\sigma_k^2$ and contributes to the sum of the squared error $\xi(n)$ by an amount $|P_k|^2$. If σ_k^2 is small and $|P_k|^2$ is not correspondingly small, a considerable control effort may be required to reduce this component of $\xi(n)$, even though it may not be significant compared to the minimum mean-square error.

For the constrained solution $\gamma \neq 0$, the steady-state value of the control effort for a convergent system is [49.56]

$$\boldsymbol{W}^{\mathrm{o}^{\mathrm{H}}}\boldsymbol{W}^{\mathrm{o}} = \sum_{k=1}^{K} \frac{\sigma_k^2 |P_k|^2}{\left(\sigma_k^2 + \gamma\right)^2}\,. \tag{49.43}$$

The first K components of the residual error are largely removed by the action of the ANC system assuming that $\sigma_k^2 \gg \gamma$, $k = 1, 2, \ldots, K$. If $\sigma_k^2 \gg \gamma$, the factor γ plays no role in the convergence of that component of $\xi(n)$. However, if $\sigma_k^2 \ll \gamma$, the control effort is limited for the k-th mode to a value $\sigma_k^2 |P_k|^2/\gamma^2$, so smaller σ_k^2 requires smaller effort.

A suitably chosen value of γ will not only improve convergence but can also prevent unreasonable values of control effort on an insignificant noise component. Moderating the control effort also considerably reduces the risk of instability due to errors in the secondary-path models [49.18]. Furthermore, the constrained algorithm is equivalent to the leaky FXLMS algorithm, which has the additional benefit of reducing the effects of numerical errors and preventing algorithm stalling.

49.4.4 Multichannel IIR Algorithm

The broadband feedforward multiple-reference/multiple-output ANC system uses adaptive IIR filters to provide longer impulse responses and feedback compensation [49.23, 59]. As illustrated in Fig. 49.14 [49.2], the controller contains two filter sections. The first section is a block of FIR filters from each reference sensor to each secondary source, while the second section implements matrix IIR filters. The feedforward section $\boldsymbol{A}(z)$ consists of elements $A_{kj}(z)$ from input $x_j(n)$ to the k-th summing node to produce $y_k(n)$, the k-th component of the output vector $\boldsymbol{y}(n)$. The transfer function matrix $\boldsymbol{B}(z)$ represents the recursive section of the controller with element $B_{ki}(z)$ from $y_i(n-1)$ to $y_k(n)$. Therefore, the secondary signal $y_k(n)$ that drives the k-th secondary source is expressed as

$$y_k(n) = \sum_{j=1}^{J} \boldsymbol{a}_{kj}^{\mathrm{T}}(n)\boldsymbol{x}_j(n) + \sum_{i=1}^{K} \boldsymbol{b}_{ki}^{\mathrm{T}}(n)\boldsymbol{y}_i(n-1)$$
$$\text{for } k = 1, 2, \ldots, K\,, \tag{49.44}$$

where $\boldsymbol{a}_{kj}(n) \equiv [a_{kj,0}(n)\, a_{kj,1}(n)\, \ldots\, a_{kj,L-1}(n)]^{\mathrm{T}}$ and $\boldsymbol{b}_{ki}(n) \equiv [b_{ki,0}(n)\, b_{ki,1}(n)\, \ldots\, b_{ki,I-1}(n)]^{\mathrm{T}}$ are the feedforward and feedback filter coefficients, respectively; $\boldsymbol{x}_j(n)$ and $\boldsymbol{y}_k(n)$ are the reference input signal vectors and output signal vectors, respectively. A multiple-reference/multiple-output filtered-U recursive LMS

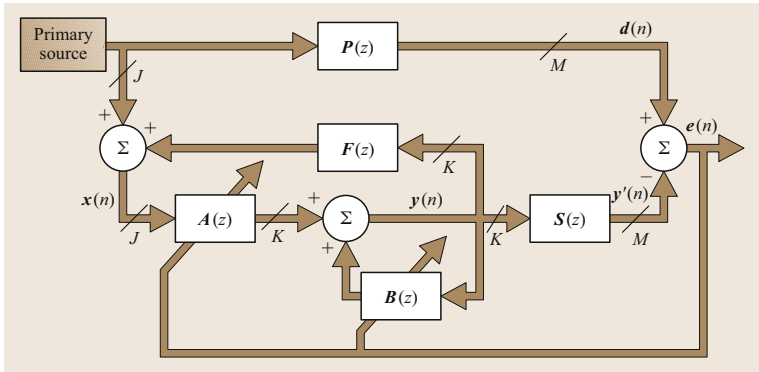


Fig. 49.14 Multichannel ANC algorithm using adaptive IIR filters

algorithm [49.59] for the IIR filter structure minimizes the sum of the M mean-square error signals:

$$\boldsymbol{a}_{kj}(n+1) = \boldsymbol{a}_{kj}(n) + \mu \sum_{m=1}^{M} \boldsymbol{x}'_{jkm}(n) e_m(n) \,, \quad (49.45)$$

$$\boldsymbol{b}_{ki}(n+1) = \boldsymbol{b}_{ki}(n) + \mu \sum_{m=1}^{M} \boldsymbol{y}'_{ikm}(n) e_m(n) \,, \quad (49.46)$$

where $\boldsymbol{x}'_{jkm}(n) \equiv \hat{s}_{mk}(n) * \boldsymbol{x}_j(n)$ and $\boldsymbol{y}'_{ikm}(n) \equiv \hat{s}_{mk}(n) * \boldsymbol{y}_i(n)$ are, respectively, the filtered reference and output signal vectors.

Multichannel adaptive IIR filtering has been successfully applied to the active control of random noise in a small reverberant room [49.23]. The matrix IIR structure results in a more-stable configuration in the presence of feedback, especially if a small leakage factor is included. The experimental results also show that far better performance is achieved by using IIR filters rather than FIR filters when the primary noise source has a lightly damped dynamic behavior.

49.4.5 Multichannel Adaptive Feedback ANC Systems

The single-channel feedback ANC system can be extended to a $K \times 1$ system that has K secondary sources and one error sensor. The reference signal $x(n)$ of the feedback ANC system is synthesized as an estimate of the primary noise

$$x(n) \equiv \hat{d}(n) = e(n) + \sum_{k=1}^{K} \hat{s}_k(n) * y_k(n) \,, \quad (49.47)$$

where $\hat{s}_k(n)$ is the impulse responses of the filter $\hat{S}_k(z)$ that models the secondary path $S_k(z)$ from the k-th secondary source to the error sensor, and the $y_k(n)$ are the secondary signals obtained from the adaptive filters $W_k(z)$.

The FXLMS algorithm is used to minimize the error signal $e(n)$ by adjusting the weight vector for each adaptive filter $W_k(z)$ according to

$$\boldsymbol{w}_k(n+1) = \boldsymbol{w}_k(n) + \mu \boldsymbol{x}'_k(n) e(n) \,, \quad k = 1, 2, \ldots, K \,, \quad (49.48)$$

where

$$\boldsymbol{x}'_k(n) \equiv \hat{s}_k(n) * \boldsymbol{x}(n) \quad (49.49)$$

is the reference signal vector filtered by the secondary-path estimate $\hat{S}_k(z)$.

49.5 Summary

This chapter has reviewed the practical aspects of ANC systems in terms of adaptive algorithms. The most widely used ANC system using the adaptive FIR filter and the FXLMS algorithm was introduced and analyzed based on single-channel broadband feedforward control. This basic system was extended to narrowband feedforward and adaptive feedback ANC systems. These single-channel ANC algorithms were then expanded to multichannel cases for controlling the noise field in an enclosure or a large-dimension duct.

References

49.1 P.A. Nelson, S.J. Elliott: *Active Control of Sound* (Academic, San Diego 1992)

49.2 S.M. Kuo, D.R. Morgan: *Active Noise Control Systems – Algorithms and DSP Implementations* (Wiley, New York 1996)

49.3 C.R. Fuller, S.J. Elliott, P.A. Nelson: *Active Control of Vibration* (Academic, San Diego 1996)

49.4 C.H. Hansen, S.D. Snyder: *Active Control of Noise and Vibration* (E&FN Spon, London 1997)

49.5 S.J. Elliott: *Signal Processing for Active Control* (Academic, San Diego 2001)

49.6 O. Tokhi, S. Veres: *Active Sound and Vibration Control: Theory and Applications* (Institute of Electrical Engineers, London 2002)

49.7 S.M. Kuo, D.R. Morgan: Active noise control: A tutorial review, Proc. IEEE **87**, 943–973 (1999)

49.8 P. Lueg: Process of silencing sound oscillations, U.S. Patent 2043416, June 9 (1936).

49.9 B. Widrow, S.D. Stearns: *Adaptive Signal Processing* (Prentice-Hall, Englewood Cliffs 1985)

49.10 S. Haykin: *Adaptive Filter Theory*, 4th edn. (Prentice Hall, Upper Saddle River 2002)

49.11 J.C. Burgess: Active adaptive sound control in a duct: A computer simulation, J. Acoust. Soc. Am. **70**, 715–726 (1981)

49.12 S.M. Kuo, W.S. Gan: *Digital Signal Processors* (Prentice Hall, Upper Saddle River 2005)

49.13 D.R. Morgan: A hierarchy of performance analysis techniques for adaptive active control of sound and vibration, J. Acoust. Soc. Am. **89**, 2362–2369 (1991)

49.14 A. Roure: Self-adaptive broadband active sound control system, J. Sound Vib. **101**, 429–441 (1985)

49.15 D.R. Morgan: An analysis of multiple correlation cancellation loops with a filter in the auxiliary path, IEEE Trans. Acoust. Speech Signal Process. **ASSP-28**, 454–467 (1980)

49.16 B. Widrow, D. Shur, S. Shaffer: On adaptive inverse control, Proc. 15th Asilomar Conf. (1981) pp. 185–189

49.17 S.J. Elliott, P.A. Nelson: Active noise control, IEEE Signal Process. Mag. **10**, 12–35 (1993)

49.18 C.C. Boucher, S.J. Elliott, P.A. Nelson: Effect of errors in the plant model on the performance of algorithms for adaptive feedforward control, IEE Proc. F Radar Signal Process. **138**, 313–319 (1991)

49.19 S.D. Snyder, C.H. Hansen: The effect of transfer function estimation errors on the filtered-X LMS algorithm, IEEE Trans. Signal Process. **42**, 950–953 (1994)

49.20 G. Long, F. Ling, J.G. Proakis: The LMS algorithm with delayed coefficient adaptation, IEEE Trans. Acoust. Speech Signal Process. **37**, 1397–1405 (1989)

49.21 S.J. Elliott, I.M. Stothers, P.A. Nelson: A multiple error LMS algorithm and its application to the active control of sound and vibration, IEEE Trans. Acoust. Speech Signal Process. **ASSP-35**, 1423–1434 (1987)

49.22 R.D. Gitlin, H.C. Meadors, S.B. Weinstein: The tap-leakage algorithm: An algorithm for the stable operation of a digital implemented, fractional adaptive spaced equalizer, Bell Syst. Tech. J. **61**, 1817–1839 (1982)

49.23 S. Laugesen, S.J. Elliott: Multichannel active control of random noise in a small reverberant room, IEEE Trans. Signal Process. **1**, 241–249 (1993)

49.24 M.M. Sondhi, D.A. Berkley: Silencing echoes on the telephone network, Proc. IEEE **68**, 948–963 (1980)

49.25 J.R. Treichler: Adaptive algorithms for infinite impulse response filters. In: *Adaptive Filters*, ed. by C.F.N. Cowan, P.M. Grant (Prentice-Hall, Englewood Cliffs 1985), Chap. 4

49.26 L.J. Eriksson, M.C. Allie, R.A. Greiner: The selection and application of an IIR adaptive filter for use in active sound attenuation, IEEE Trans. Acoust. Speech Signal Process. **ASSP-35**, 433–437 (1987)

49.27 P.L. Feintuch: An adaptive recursive LMS filter, Proc. IEEE **64**, 1622–1624 (1976)

49.28 L.J. Eriksson: Development of the filtered-U algorithm for active noise control, J. Acoust. Soc. Am. **89**, 257–265 (1991)

49.29 L.J. Eriksson, M.C. Allie, C.D. Bremigan, J.A. Gilbert: Active noise control on systems with time-varying sources and parameters, Sound Vib. **23**, 16–21 (1989)

49.30 S.J. Elliott, P. Darlington: Adaptive cancellation of periodic, synchronously sampled interference, IEEE Trans. Acoust. Speech Signal Process. **ASSP-33**, 715–717 (1985)

49.31 B. Chaplin: The cancellation of repetitive noise and vibration, Proc. Inter-noise **1980**, 699–702 (1980)

49.32 B. Widrow, J.R. Glover, J.M. McCool, J. Kaunitz, C.S. Williams, R.H. Hern, J.R. Zeidler, E. Dong, R.C. Goodlin: Adaptive noise canceling: principles and applications, Proc. IEEE **63**, 1692–1716 (1975)

49.33 P. Darlington, S.J. Elliott: Stability and adaptively controlled systems – a graphical approach, Proc. ICASSP **1987**, 399–402 (1987)

49.34 P. Darlington, S.J. Elliott: Synchronous adaptive filters with delayed coefficient adaptation, Proc. ICASSP **1988**, 2586–2589 (1988)

49.35 D.R. Morgan, J. Thi: A multitone pseudocascade filtered-X LMS adaptive notch filter, IEEE Trans. Signal Process. **41**, 946–956 (1993)

49.36 J.R. Glover Jr.: Adaptive noise canceling applied to sinusoidal interferences, IEEE Trans. Acoust. Speech Signal Process. **ASSP-25**, 484–491 (1977)

49.37 E. Ziegler Jr.: Selective active cancellation system for repetitive phenomena, U.S. Patent 4878188, Oct. 31 (1989)

49.38 D.R. Morgan, C. Sanford: A control theory approach to the stability and transient analysis of the filtered-X LMS adaptive notch filter, IEEE Trans. Signal Process. **40**, 2341–2346 (1992)

49.39 X. Kong, S.M. Kuo: Analysis of asymmetric out-of-band overshoot in narrowband active noise control systems, IEEE Trans. Speech Audio Process. **7**, 587–591 (1999)

49.40 S.M. Kuo, M. Ji: Passband disturbance reduction in periodic active noise control systems, IEEE Trans. Speech Audio Process. **4**, 96–103 (1996)

49.41 D. P. Pfaff, N. S. Kapsokavathis, N. A. Parks: Method for actively attenuating engine generated noise, U.S. Patent 5146505, Sept. 8 (1992).

49.42 S.M. Kuo, A.B. Puvvala: Effects of frequency separation in periodic active noise control systems, IEEE Trans. Speech Audio Process. **14**, 1857–1866 (2006)

49.43 Y. Yuan, N. S. Kapsokavathis, K. Chen, S. M. Kuo: Active noise control system, U.S. Patent 5359662, Oct. 25 (1994).

49.44 S. M. Kuo, M. J. Ji, M. K. Christensen, R. A. Herold: Indirectly sensed signal processing in active periodic acoustic noise cancellation, U.S. Patent 5502770, Mar. 26 (1996).

49.45 P. D. Hill: Active acoustic attenuation system for reducing tonal noise in rotating equipment, U.S. Patent 5010576, Apr. 23 (1991).

49.46 S.M. Kuo, M.J. Ji: Development and analysis of an adaptive noise equalizer, IEEE Trans. Speech Audio Process. **3**, 217–222 (1995)

49.47 S.M. Kuo, Y. Yang: Broadband Adaptive Noise Equalizer, IEEE Signal Process. Lett. **3**, 234–235 (1996)

49.48 L.J. Eriksson: Recursive algorithms for active noise control, Proc. Int. Symp. Active Control of Sound Vib. (1991) pp. 137–146

49.49 S.J. Elliott, T.J. Sutton: Performance of feedforward and feedback systems for active control, IEEE Trans. Speech Audio Process. **4**, 214–223 (1996)

49.50 Y. Song, Y. Gong, S.M. Kuo: A robust hybrid feedback active noise cancellation headset, IEEE Trans. Speech Audio Process. **13**, 607–617 (2005)

49.51 S.M. Kuo, S. Mitra, W.S. Gan: Active noise control system for headphone applications, IEEE Trans. Contr. Syst. Technol. **14**, 331–335 (2006)

49.52 D.C. Swanson: Active noise attenuation using a self-tuning regulator as the adaptive control algorithm, Proc. Inter-noise **1989**, 467–470 (1989)

49.53 P.A. Nelson, A.R.D. Curtis, S.J. Elliott, A.J. Bullmore: The active minimization of harmonic enclosed sound fields, Part I: Theory, J. Sound Vib. **117**, 1–13 (1987)

49.54 A.J. Bullmore, P.A. Nelson, A.R.D. Curtis, S.J. Elliott: The active minimization of harmonic enclosed sound fields, Part II: A computer simulation, J. Sound Vib. **117**, 15–33 (1987)

49.55 S.J. Elliott, A.R.D. Curtis, A.J. Bullmore, P.A. Nelson: The active minimization of harmonic enclosed sound fields, Part III: Experimental verification, J. Sound Vib. **117**, 35–58 (1987)

49.56 S.J. Elliott, C.C. Boucher, P.A. Nelson: The behavior of a multiple channel active control system, IEEE Trans. Signal Process. **40**, 1041–1052 (1992)

49.57 P.A. Nelson, A.R.D. Curtis, S.J. Elliott, A.J. Bullmore: The minimum power output of free field point sources and the active control of sound, J. Sound Vib. **116**, 397–414 (1987)

49.58 S.J. Elliott, C.C. Boucher: Interaction between multiple feedforward active control systems, IEEE Trans. Speech Audio Process. **2**, 521–530 (1994)

49.59 D.E. Melton, R.A. Greiner: Adaptive feedforward multiple-input, multiple-output active noise control, Proc. ICASSP **1992**, 229–232 (1992)

47. Adaptive Beamforming and Postfiltering

S. Gannot, I. Cohen

In this chapter, we explore many of the basic concepts of array processing with an emphasis on adaptive beamforming for speech enhancement applications. We begin in Sect. 47.1 by formulating the problem of microphone array in a noisy and reverberant environment. In Sect. 47.2, we derive the frequency-domain linearly constrained minimum-variance (LCMV) beamformer, and its generalized sidelobe canceller (GSC) variant. The GSC components are explored in Sect. 47.3, and several commonly used special cases of these blocks are presented. As the GSC structure necessitates an estimation of the speech related acoustical transfer functions (ATFs), several alternative system identification methods are addressed in Sect. 47.4. Beamformers often suffer from sensitivity to signal mismatch. We analyze this phenomenon in Sect. 47.5 and explore several cures to this problem. Although the GSC beamformer yields a significant improvement in speech quality, when the noise field is spatially incoherent or diffuse, the noise reduction is insufficient and additional postfiltering is normally required. In Sect. 47.6, we present multimicrophone postfilters, based on either minimum mean-squared error (MMSE) or log-spectral amplitude estimate criteria. An interesting relation between the GSC and the Wiener filter is derived in this Section as well. In Sect. 47.7, we analyze the performance of the transfer-function GSC (TF-GSC), and in Sect. 47.8 demonstrate the advantage of multichannel postfiltering over single-channel postfiltering in nonstationary noise conditions.

Over the last four decades, array processing has become a well-established discipline, see e.g., [47.1–14]. In the mid 1980s, array processing and beamforming methods were adopted by the speech community to deal with data received by microphone arrays. Since then, beamforming techniques for microphone arrays have been used in many applications, such as speaker separation, speaker localization, speech dereverberation, acoustic echo cancellation, and speech enhancement.

Adaptive beamforming for speech signals requires particular consideration of problems that are specific to speech signals and to the acoustic environment. The speech signal is wide-band, highly nonstationary, and has a very wide dynamic range. An acoustic enclosure is usually modeled as a filter with very long impulse response due to multiple reflections from the room walls. In a typical office, the length of the filters may reach several thousand taps. Furthermore, the impulse response is often time varying due to speaker and objects movements.

The term beamforming refers to the design of a spatiotemporal filter. Broadband arrays comprise a set of filters, applied to each received microphone signal, followed by a summation operation. The main objective of the beamformer is to extract a desired signal, impinging on the array from a specific position, out of noisy measurements thereof. Usually, the interference signals occupy the same frequency band as the desired signal, rendering temporal-only filtering useless. The simplest structure is the delay-and-sum beamformer, which first compensates for the relative delay between distinct microphone signals and then sums the steered signal to form a single output. This beamformer, which is still widely used, can be very effective in mitigating noncoherent, i. e., spatially white, noise sources, provided that the number of microphones is relatively high. However, if the noise source is coherent, the noise reduction (NR) is strongly dependent on the direction of arrival of the noise signal. Consequently, the performance of the delay and sum beamformer in reverberant environments is often insufficient. *Jan* and *Flanagan* [47.15, 16] and *Rabinkin* et al. [47.17] extended the delay and sum concept by introducing the filter-and-sum beamformer. This structure, designed for multipath environments, namely reverberant enclosures, replaces the simpler delay compensator with a matched filter.

The array beam pattern can generally be designed to have a specified response. This can be done by properly setting the values of the multichannel filters' weights. However, the application of data-independent design methods is very limited in dynamic acoustical environments. Statistically optimal beamformers are designed based on the statistical properties of the desired and interference signals. In general, they aim at enhancing the desired signal, while rejecting the interference signal. Several criteria can be applied in the design of the beamformer, e.g., maximum signal-to-noise ratio (MSNR), minimum mean-squared error (MMSE), and linearly constrained minimum variance (LCMV). A summary of several design criteria can be found in [47.5, 7].

Beamforming methods use the signals' statistics (at least second-order statistics), which is usually not available and should be estimated from the data. Moreover, the acoustical environment is time varying, due to talker and objects movements, and abrupt changes in the noise characteristics (e.g., passing cars). Hence, adaptation mechanisms are required. An adaptive counterpart of each of the prespecified design criteria can be derived. Early contributions to the field of adaptive beamformers design can be attributed to *Sondhi* and *Elko* [47.18], to *Kaneda* and *Ohga* [47.19], and to *Van Compernolle* [47.20]. *Kellermann* [47.21] addressed the problem of joint echo cancellation and NR by incorporating echo cancellers into the beamformer design. *Nordholm* et al. [47.22, 23] used microphone arrays in a car environment, and designed a beamformer employing calibration signals to enhance the obtained performance. *Martin* [47.24] analyzed beamforming techniques for small microphone arrays. Many other applications of microphone arrays such as hearing aids, blind source separation (BSS), and dereverberation are addressed elsewhere in this handbook.

The minimization of the mean-squared error (MSE) in the context of array processing leads to the well-known multichannel Wiener filter [47.25]. *Doclo* and *Moonen* [47.26–28] proposed an efficient implementation of the Wiener filter based on the generalized singular-value decomposition (GSVD) of the microphone data matrix. This method yields an optimal estimation (in the MMSE sense) of the desired signal component of one of the microphone signals. The authors further proposed efficient schemes for recursive update of the GSVD. An optional, adaptive noise cancellation postfiltering stage is proposed as well. In that scheme, in addition to the optimal estimation of the desired speech signal, an optimal noise channel is also estimated. This estimated noise component can be used as a reference noise signal (similar to the one used in [47.25]), to further enhance the speech signal. *Spriet* et al. [47.29] proposed a subband implementation of the GSVD-based scheme, and *Rombouts* and

Moonen [47.30, 31] proposed to apply the efficient QR decomposition to the problem at hand.

In many adaptive array schemes the acoustical transfer-function (ATF) relating the speech source and the microphone should be known in advance, or at least estimated from the received data (note that in case of delay-only propagation, the acoustical transfer function reduces to a steering vector, consisting of phase-only components.) In contrast, the multichannel Wiener filter is uniquely based on estimates of the second-order statistics of the recorded noisy signal and the noise signal (estimated during noise-only segments), and does not make any a priori assumptions about the signal model. Unfortunately, as pointed by *Chen* et al. [47.32], the Wiener filter, which is optimal in the MMSE sense, cannot guarantee undistorted speech signal at its output. This drawback can however be mitigated by modifying the MMSE criterion to control the amount of imposed speech distortion. A method that employs this modification is presented in [47.33, 34]. It is also shown that the ATFs information (only a simple delay-only case is presented in the contributions) can be incorporated into the Wiener filter scheme (called the spatially preprocessed Wiener filter), resulting in an improved performance. The Wiener filter and its application to speech enhancement is addressed in a separate chapter of this handbook (6; 43).

In this chapter, we concentrate on a different adaptive structure based on the LCMV criterion. The LCMV beamformer, proposed by *Frost* [47.35], is aiming at minimizing the output power under linear constraints on the response of the array towards the desired speech signal. Frost proposed an adaptive scheme, which is based on a constrained least-mean-square (LMS)-type adaptation (for the LMS algorithm please refer to [47.25]). To avoid this constrained adaptation, *Griffiths* and *Jim* [47.36] proposed the GSC structure, which separates the output power minimization and the application of the constraint. The GSC structure is based on the assumption that the different sensors receive a delayed version of the desired signal, and therefore we refer to it as the delay generalized sidelobe canceller (D-GSC). The GSC structure was rederived in the frequency domain, and extended to deal with, the more-complicated general ATFs case by *Affes* and *Grenier* [47.37] and later by *Gannot* et al. [47.38]. This frequency-domain version, which takes into account the reverberant nature of the enclosure, was nicknamed the transfer-function generalized sidelobe canceller (TF-GSC). The GSC comprises three blocks: a fixed beamformer (FBF), which aligns the desired signal components, a blocking matrix (BM), which blocks the desired speech components resulting in reference noise signals, and a multichannel adaptive noise canceller (ANC), which eliminates noise components that leak through the sidelobes of the FBF.

Nordholm and *Leung* [47.39] analyze the limits of the obtainable NR of the GSC in an isotropic noise field. *Bitzer* et al. address the problem in [47.40, 41] and [47.42]. In [47.40], the authors derive an expression for the NR as a function of the noise field and evaluate the degradation as a function of the reverberation time (T_{60}). The special two-microphone case is treated in [47.41]. The additional NR due to the ANC branch of the GSC, implemented by a closed-form Wiener filter rather than the adaptive Widrow least-mean-square (LMS) procedure, is presented in [47.42]. The frequency-band nested subarrays structure is presented and its NR is theoretically analyzed by *Marro* et al. [47.43]. A more-complex dual GSC structure employing calibration signals was suggested and analyzed by *Nordholm* et al. [47.44]. *Huarng* and *Yeh* [47.45] addressed the distortion issue by evaluating the desired signal leakage into the reference noise branch of the GSC structure. However, the delay-only ATFs assumption is imposed and the expected degradation due to pointing errors alone is evaluated. The performance degradation due to constraining the Wiener filters to an finite impulse response (FIR) structure is demonstrated by *Nordholm* et al. in [47.46]. The resulting performance limits of the GSC structure strongly depend on the cross-correlation between the sensors' signals induced by the noise field, as shown in the above references and by *Cox* [47.47].

47.1 Problem Formulation

Consider an array of M sensors in a noisy and reverberant environment. The received signals generally include three components. The first is a desired speech signal, the second is some stationary interference signal, and the third is some nonstationary (transient) noise component. Our goal is to reconstruct the speech component from the received signals. Let $s(t)$ denote the desired source signal, let $a_m(t)$ represent the room impulse response (RIR) of the m-th sensor to the desired source, and let $n_m(t)$ denote the noise component at the m-th sensor.

The observed signal at the m-th sensor ($m = 1, \ldots, M$) is given by

$$\begin{aligned} z_m(t) &= a_m(t) * s(t) + n_m(t) \\ &= a_m(t) * s(t) + n_m^{\mathrm{s}}(t) + n_m^{\mathrm{t}}(t)\,, \end{aligned} \tag{47.1}$$

where $n_m^{\mathrm{s}}(t)$ and $n_m^{\mathrm{t}}(t)$ represent the stationary and nonstationary noise components at the m-th sensor, respectively, and $*$ denotes convolution. We assume that both noise components may comprise coherent (directional) noise component and diffused noise component.

The observed signals are divided in time into overlapping frames by the application of a window function and analyzed using the short-time Fourier transform (STFT). Assuming time-invariant transfer functions, we have in the time–frequency domain

$$\begin{aligned} Z_m(k,\ell) &\approx A_m(k)S(k,\ell) + N_m(k,\ell) \\ &\approx A_m(k)S(k,\ell) + N_m^{\mathrm{s}}(k,\ell) + N_m^{\mathrm{t}}(k,\ell)\,, \end{aligned} \tag{47.2}$$

where ℓ is the frame index and $k = 1, 2, \ldots, K$ represents the frequency bin index. (The equality in (47.2) is only justified for segments which are longer than the RIR length. Since RIRs tend to be very long, the conditions allowing for this representation to hold cannot be exactly met. We assume, however, that the STFT relation is a reasonable approximation.) $Z_m(k,\ell)$, $S(k,\ell)$, $N_m(k,\ell)$, $N_m^{\mathrm{s}}(k,\ell)$, and $N_m^{\mathrm{t}}(k,\ell)$ are the STFT of the respective signals. $A_m(k)$ is the ATF relating the speech source with the m-th sensor. The vector formulation of the equation set (47.2) is

$$\begin{aligned} \boldsymbol{Z}(k,\ell) &= \boldsymbol{A}(k)S(k,\ell) + \boldsymbol{N}(k,\ell) \\ &= \boldsymbol{A}(k)S(k,\ell) + \boldsymbol{N}_{\mathrm{s}}(k,\ell) + \boldsymbol{N}_{\mathrm{t}}(k,\ell)\,, \end{aligned} \tag{47.3}$$

where

$$\begin{aligned} \boldsymbol{Z}(k,\ell) &= \big(Z_1(k,\ell)\ \ Z_2(k,\ell)\ \ \cdots\ \ Z_M(k,\ell)\big)^{\mathrm{T}}\,, \\ \boldsymbol{A}(k) &= \big(A_1(k)\ \ A_2(k)\ \ \cdots\ \ A_M(k)\big)^{\mathrm{T}}\,, \\ \boldsymbol{N}(k,\ell) &= \big(N_1(k,\ell)\ \ N_2(k,\ell)\ \ \cdots\ \ N_M(k,\ell)\big)^{\mathrm{T}}\,, \\ \boldsymbol{N}_{\mathrm{s}}(k,\ell) &= \big(N_1^{\mathrm{s}}(k,\ell)\ \ N_2^{\mathrm{s}}(k,\ell)\ \ \cdots\ \ N_M^{\mathrm{s}}(k,\ell)\big)^{\mathrm{T}}\,, \\ \boldsymbol{N}_{\mathrm{t}}(k,\ell) &= \big(N_1^{\mathrm{t}}(k,\ell)\ \ N_2^{\mathrm{t}}(k,\ell)\ \ \cdots\ \ N_M^{\mathrm{t}}(k,\ell)\big)^{\mathrm{T}}. \end{aligned}$$

47.2 Adaptive Beamforming

Frost [47.35] proposed a beamformer that relies on the assumption that the ATFs between the desired source and the array of sensors can be uniquely determined by gain and delay values. In this section, we follow Frost's approach in the STFT domain and derive a beamforming algorithm for the arbitrary ATF case. We first obtain a closed form of the LCMV beamformer, and subsequently derive an adaptive solution. The outcome is a constrained LMS-type algorithm. We proceed, following the seminal work of *Griffiths* and *Jim* [47.36], with the formulation of an unconstrained adaptive solution namely, the transfer-function generalized sidelobe canceller (TF-GSC). We initially assume that the ATFs are known. Later, in Sect. 47.4, we present several alternatives for estimating the ATFs.

47.2.1 Frequency-Domain Frost Algorithm

Optimal Solution

Let $W_m^*(k,\ell)\,;\, m = 1, \ldots, M$ denote a set of M filters, and define

$$\boldsymbol{W}^{\mathrm{H}}(k,\ell) = \big(W_1^*(k,\ell)\ \ W_2^*(k,\ell)\ \ \cdots\ \ W_M^*(k,\ell)\big)\,,$$

where the superscript H denotes conjugation transpose. A filter-and-sum beamformer, depicted in Fig. 47.1, is realized by filtering each sensor signal by $W_m^*(k,\ell)$ and summing the outputs,

$$\begin{aligned} Y(k,\ell) &= \boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{Z}(k,\ell) \\ &= \boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{A}(k)S(k,\ell) + \boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{N}_{\mathrm{s}}(k,\ell) \\ &\quad + \boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{N}_{\mathrm{t}}(k,\ell) \\ &\triangleq Y_{\mathrm{s}}(k,\ell) + Y_{n,\mathrm{s}}(k,\ell) + Y_{n,\mathrm{t}}(k,\ell)\,, \end{aligned} \tag{47.4}$$

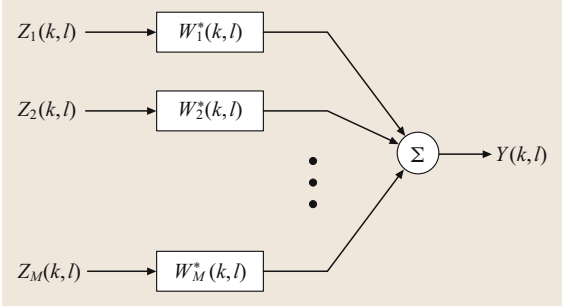


Fig. 47.1 Filter-and-sum beamformer

where $Y_{\mathrm{s}}(k,\ell)$ is the signal component and $Y_{n,\mathrm{s}}(k,\ell)$ and $Y_{n,\mathrm{t}}(k,\ell)$ are the stationary and nonstationary noise components, respectively. The output power of the beamformer is given by

$$\begin{aligned} &E\{Y(k,\ell)Y^*(k,\ell)\} \\ &= E\{\boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{Z}(k,\ell)\boldsymbol{Z}^{\mathrm{H}}(k,\ell)\boldsymbol{W}(k,\ell)\} \\ &= \boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{\phi}_{\boldsymbol{ZZ}}(k,\ell)\boldsymbol{W}(k,\ell)\,, \end{aligned}$$

where $\boldsymbol{\phi}_{\boldsymbol{ZZ}}(k,\ell) \triangleq E\{\boldsymbol{Z}(k,\ell)\boldsymbol{Z}^{\mathrm{H}}(k,\ell)\}$ is the power spectral density (PSD) matrix of the received signals. We want to minimize the output power subject to the following constraint on $Y_{\mathrm{s}}(k,\ell)$:

$$\begin{aligned} Y_{\mathrm{s}}(k,\ell) &= \boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{A}(k)S(k,\ell) \\ &= \mathcal{F}^*(k,\ell)S(k,\ell)\,, \end{aligned}$$

where $\mathcal{F}(k,\ell)$ is some prespecified filter, usually a simple delay. Without loss of generality we assume hereinafter that $\mathcal{F}(k,\ell)=1$. Hence, the minimization problem can be stated as

$$\begin{aligned} &\min_{\boldsymbol{W}} \left\{\boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{\phi}_{\boldsymbol{ZZ}}(k,\ell)\boldsymbol{W}(k,\ell)\right\} \\ &\text{subject to} \quad \boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{A}(k)=1\,. \end{aligned} \tag{47.5}$$

The minimization problem (47.5) is demonstrated in Fig. 47.2. The point where the equipower contours are tangent to the constraint plane is the optimum vector of beamforming filters. The perpendicular $\boldsymbol{F}(k)$ from the origin to the constraint plane will be calculated in the next section.

To solve (47.5) we first define the complex Lagrangian,

$$\begin{aligned} \mathcal{L}(\boldsymbol{W}) &= \boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\boldsymbol{W}(k,\ell) \\ &\quad +\lambda[\boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{A}(k)-1] \\ &\quad +\lambda^*[\boldsymbol{A}^{\mathrm{H}}(k,\ell)\boldsymbol{W}(k)-1]\,, \end{aligned} \tag{47.6}$$

where λ is a Lagrange multiplier. Setting the derivative with respect to $\boldsymbol{W}^*$ to 0 [47.48] yields

$$\nabla_{\boldsymbol{W}^*}\mathcal{L}(W)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\boldsymbol{W}(k,\ell)+\lambda\boldsymbol{A}(k)=0.$$

Now, recalling the constraint in (47.5), we obtain the LCMV optimal filter

$$\boldsymbol{W}^{\mathrm{LCMV}}(k,\ell) = \frac{\boldsymbol{\Phi}_{\boldsymbol{ZZ}}^{-1}(k,\ell)\boldsymbol{A}(k)}{\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}^{-1}(k,\ell)\boldsymbol{A}(k)}\,. \tag{47.7}$$

This closed-form solution is difficult to implement, and is not suitable for time-varying environments. Therefore we often have to resort to an adaptive solution, which is derived in the sequel.

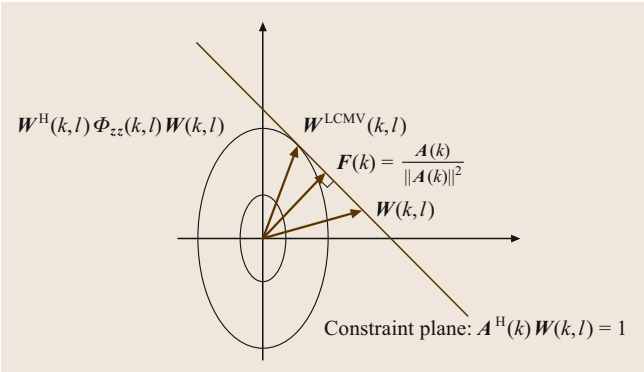


Fig. 47.2 Constrained minimization

It is interesting to show the equivalence between the LCMV solution (47.7) and the MSNR beamformer [47.7], which is obtained from

$$\max_{\boldsymbol{W}} \frac{|\boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{A}(k)|^2}{\boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell)\boldsymbol{W}(k,\ell)}\,. \tag{47.8}$$

The well-known solution to (47.8) is the (colored-noise) matched filter

$$\boldsymbol{W}(k,\ell) \propto \boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}(k,\ell)\boldsymbol{A}(k)\,.$$

If the array response is constrained to fulfil $\boldsymbol{W}^{\mathrm{H}}(k,\ell)\boldsymbol{A}(k)=1$, i. e., no distortion in the desired direction, we have

$$\boldsymbol{W}^{\mathrm{MSNR}}(k,\ell) = \frac{\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}(k,\ell)\boldsymbol{A}(k)}{\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}(k,\ell)\boldsymbol{A}(k)}\,. \tag{47.9}$$

Using (47.3) it can be verified that

$$\begin{aligned} \boldsymbol{\Phi}_{\boldsymbol{ZZ}} &= \phi_{\mathrm{ss}}(k,\ell)\boldsymbol{A}(k)\boldsymbol{A}^{\mathrm{H}}(k) \\ &\quad +\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)+\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{t}}\boldsymbol{N}_{\mathrm{t}}}(k,\ell) \\ &= \phi_{\mathrm{ss}}(k,\ell)\boldsymbol{A}(k)\boldsymbol{A}^{\mathrm{H}}(k)+\boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell)\,, \end{aligned} \tag{47.10}$$

where $\boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell) \triangleq \boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)+\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{t}}\boldsymbol{N}_{\mathrm{t}}}(k,\ell)$, the overall noise PSD matrix. Using the matrix inversion lemma, it is shown in Appendix 47.B that

$$\boldsymbol{W}^{\mathrm{LCMV}}(k,\ell) = \frac{\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}(k,\ell)\boldsymbol{A}(k)}{\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}(k,\ell)\boldsymbol{A}(k)}\,. \tag{47.11}$$

This solution is identical to the solution of the MSNR beamformer.

While both methods are shown to be equal, provided that the ATFs $\boldsymbol{A}(k)$ are known, their behavior in the case of unknown ATFs is different. Analysis of these differences is given by *Cox* [47.47].

Note also that, due to the nonstationary noise component, the term $\boldsymbol{\Phi}_{NN}(k,\ell)$ depends on the frame index. This time dependence is one of the major factors leading to performance degradation in beamforming. We address this problem by introducing the multichannel postfilter in Sect. 47.6.

Adaptive Solution

Consider the following steepest-descent adaptive algorithm:

$$\begin{aligned}&\boldsymbol{W}(k,\ell+1)\\&=\boldsymbol{W}(k,\ell)-\mu\nabla_{\boldsymbol{W}^*}\mathcal{L}(k,\ell)\\&=\boldsymbol{W}(k,\ell)-\mu\left[\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\boldsymbol{W}(k,\ell)+\lambda\boldsymbol{A}(k)\right]\,.\end{aligned}$$

Imposing the look-direction constraint on $\boldsymbol{W}(\ell+1,k)$ yields

$$\begin{aligned}1&=\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{W}(k,\ell+1)\\&=\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{W}(k,\ell)-\mu\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\boldsymbol{W}(k,\ell)\\&\quad-\mu\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{A}(k)\lambda\,.\end{aligned}$$

Solving for the Lagrange multiplier and applying further rearrangement of terms yields:

$$\begin{aligned}&\boldsymbol{W}(k,\ell+1)\\&=P(k)\boldsymbol{W}(k,\ell)-\mu P(k)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\boldsymbol{W}(k,\ell)+\boldsymbol{F}(k)\,,\end{aligned}\tag{47.12}$$

where

$$P(k)\triangleq\mathbf{I}-\frac{\boldsymbol{A}(k)\boldsymbol{A}^{\mathrm{H}}(k)}{\|\boldsymbol{A}(k)\|^2}\tag{47.13}$$

and

$$\boldsymbol{F}(k)\triangleq\frac{\boldsymbol{A}(k)}{\|\boldsymbol{A}(k)\|^2}.\tag{47.14}$$

Further simplification can be obtained by replacing $\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)$ by its instantaneous estimator, $\boldsymbol{Z}(k,\ell)\boldsymbol{Z}^{\mathrm{H}}(k,\ell)$, and recalling (47.4). We finally obtain,

$$\begin{aligned}&\boldsymbol{W}(k,\ell+1)\\&=P(k)[\boldsymbol{W}(k,\ell)-\mu\boldsymbol{Z}(k,\ell)Y^*(k,\ell)]+\boldsymbol{F}(k)\,.\end{aligned}$$

The entire algorithm is summarized in Table 47.1.

Table 47.1 Frequency-domain Frost algorithm

$\boldsymbol{W}(\ell=0,k)=\boldsymbol{F}(k)$
$\boldsymbol{W}(t+1,k)=P(k)\left[\boldsymbol{W}(k,\ell)-\mu\boldsymbol{Z}(k,\ell)Y^*(k,\ell)\right]+\boldsymbol{F}(k)$
$\ell=0,1,\ldots$
[$P(k)$ and $\boldsymbol{F}(k)$ are defined by (47.13) and (47.14)].

47.2.2 Frequency-Domain Generalized Sidelobe Canceller

In [47.36], *Griffiths* and *Jim* considered the case where each ATF is a delay element (with gain). They obtained an unconstrained adaptive enhancement algorithm, using the same linear constraint imposed by *Frost* [47.35]. The unconstrained algorithm is more tractable, reliable, and computationally more efficient in comparison with its constrained counterpart. In the adaptive solution Section, we obtained an adaptive algorithm for the case where each ATF is represented by an arbitrary linear time-invariant system. We now repeat the arguments of Griffiths and Jim for the arbitrary ATFs case, and derive an unconstrained adaptive enhancement algorithm. A detailed description can be found in [47.38].

Derivation

Consider the null space of $\boldsymbol{A}(k)$, defined by

$$\mathcal{N}(k)\triangleq\{\boldsymbol{W}\,|\,\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{W}=0\}\,.$$

The constraint hyperplane,

$$\Lambda(k)\triangleq\{\boldsymbol{W}\,|\,\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{W}=1\}$$

is parallel to $\mathcal{N}(k)$. In addition, let

$$\mathcal{R}(k)\triangleq\{\kappa\boldsymbol{A}(k)\,|\,\text{for any real }\kappa\}$$

be the column space. By the fundamental theorem of linear algebra [47.49], $\mathcal{R}(k)\perp\mathcal{N}(k)$. In particular, $\boldsymbol{F}(k)$ is perpendicular to $\mathcal{N}(k)$, since $\boldsymbol{F}(k)=\frac{1}{\|\boldsymbol{A}(k)\|^2}\boldsymbol{A}(k)\in\mathcal{R}(k)$. Furthermore,

$$\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{F}(k)=\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{A}(k)(\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{A}(k))^{-1}=1\,.$$

Thus, $\boldsymbol{F}(k)\in\Lambda(k)$ and $\boldsymbol{F}(k)\perp\Lambda(k)$. Hence, $\boldsymbol{F}(k)$ is the perpendicular from the origin to the constraint hyperplane, $\Lambda(k)$. The matrix $P(k)$, defined in (47.13), is the projection matrix to the null space of $\boldsymbol{A}(k)$, $\mathcal{N}(k)$.

A vector in linear space can be uniquely split into a sum of two vectors in mutually orthogonal subspaces [47.49]. Hence,

$$\boldsymbol{W}(k,\ell)=\boldsymbol{W}_0(k,\ell)-\boldsymbol{V}(k,\ell)\,,\tag{47.15}$$

where $\boldsymbol{W}_0(k,\ell)\in\mathcal{R}(k)$ and $-\boldsymbol{V}(k,\ell)\in\mathcal{N}(k)$. By the definition of $\mathcal{N}(k)$,

$$\boldsymbol{V}(k,\ell)=\mathcal{H}(k)\boldsymbol{G}(k,\ell)\,,\tag{47.16}$$

where $\mathcal{H}(k)$ is a matrix such that its columns span the null space of $\boldsymbol{A}(k)$, i. e.,

$$\boldsymbol{A}^{\mathrm{H}}(k)\mathcal{H}(k)=0\,,\qquad\operatorname{rank}\{\mathcal{H}(k)\}\le M-1\,,\tag{47.17}$$

where $\mathcal{H}(k)$ is usually called a BM (blocking matrix). The outputs of the BM will be denoted, for reasons that will be clear in the sequel, noise reference signals $\boldsymbol{U}(k,\ell)$, defined as

$$\boldsymbol{U}(k,\ell) = \mathcal{H}^{\mathrm{H}}(k)\boldsymbol{Z}(k,\ell)\,, \tag{47.18}$$

where

$$\boldsymbol{U}(k,\ell) = \big(U_2(k,\ell)\;\; U_3(k,\ell)\;\; \ldots\;\; U_M(k,\ell)\big)^{\mathrm{T}}\,.$$

The vector $\boldsymbol{G}(k,\ell)$ is a rank$\{\mathcal{H}(k)\}\times 1$ vector of adjustable filters. We assume hereinafter that rank$\{\mathcal{H}(k)\} = M-1$. Hence, the set of filters is defined as

$$\boldsymbol{G}(k,\ell) = \big(G_2(k,\ell)\;\; G_3(k,\ell)\;\; \ldots\;\; G_M(k,\ell)\big)^{\mathrm{T}}\,. \tag{47.19}$$

By the geometrical interpretation of Frost's algorithm,

$$\boldsymbol{W}_0(k,\ell) = \boldsymbol{F}(k) = \frac{\boldsymbol{A}(k)}{\|\boldsymbol{A}(k)\|^2} \tag{47.20}$$

(Recall that $\boldsymbol{F}(k)$ is the perpendicular from the origin to the constraint hyperplane, $\Lambda(k)$.) Now, using (47.4), (47.15), and (47.16) we obtain

$$Y(k,\ell) = Y_{\mathrm{FBF}}(k,\ell) - Y_{\mathrm{ANC}}(k,\ell)\,, \tag{47.21}$$

where

$$\begin{aligned} Y_{\mathrm{FBF}}(k,\ell) &= \boldsymbol{W}_0^{\mathrm{H}}(k,\ell)\boldsymbol{Z}(k,\ell) \\ Y_{\mathrm{ANC}}(k,\ell) &= \boldsymbol{G}^{\mathrm{H}}(k,\ell)\mathcal{H}^{\mathrm{H}}(k)\boldsymbol{Z}(k,\ell)\,. \end{aligned} \tag{47.22}$$

The output of the constrained beamformer is a difference of two terms, both operating on the input signal $\boldsymbol{Z}(k,\ell)$. The first term, $Y_{\mathrm{FBF}}(k,\ell)$, utilizes only fixed components (which depend on the ATFs), so it can be viewed as a FBF. The FBF coherently sums the desired speech components, while in general it destructively sums the noise components. Hence, it is expected that the signal-to-noise ratio (SNR) at the FBF output will be higher than the input SNR. However, this result cannot be guaranteed. We will elaborate on this issue while discussing the performance analysis in Sect. 47.7.

We now examine the second term $Y_{\mathrm{ANC}}(k,\ell)$. Note that

$$\begin{aligned} \boldsymbol{U}(k,\ell) &= \mathcal{H}^{\mathrm{H}}(k)\boldsymbol{Z}(k,\ell) \\ &= \mathcal{H}^{\mathrm{H}}(k)[\boldsymbol{A}(k)S(k,\ell) \\ &\quad + \boldsymbol{N}_{\mathrm{s}}(k,\ell) + \boldsymbol{N}_{\mathrm{t}}(k,\ell)] \\ &= \mathcal{H}^{\mathrm{H}}(k)[\boldsymbol{N}_{\mathrm{s}}(k,\ell) + \boldsymbol{N}_{\mathrm{t}}(k,\ell)]\,. \end{aligned} \tag{47.23}$$

The last transition is due to (47.17). It is worth mentioning that, when a perfect BM is applied, $\boldsymbol{U}(k,\ell)$ indeed contains only noise components. In general, however, $\mathcal{H}^{\mathrm{H}}(k)\boldsymbol{A}(k,\ell) \neq 0$, hence desired speech components may leak into the noise reference signals. If the speech component is indeed completely eliminated (blocked) by $\mathcal{H}(k)$, $Y_{\mathrm{ANC}}(k,\ell)$ becomes a pure noise term. The residual noise term in $Y_{\mathrm{FBF}}(k,\ell)$ can then be reduced by properly adjusting the filters $\boldsymbol{G}(k,\ell)$, using the minimum output power criterion. This minimization problem is in fact the classical multichannel noise cancellation problem. An adaptive LMS solution to the problem was proposed by *Widrow* [47.25].

To summarize, the beamformer is comprised of three parts. An FBF $\boldsymbol{W}_0$, which aligns the desired signal components, a BM $\mathcal{H}(k)$, which blocks the desired speech components resulting in the reference noise signals $\boldsymbol{U}(k,\ell)$, and a multichannel ANC $\boldsymbol{G}(k,\ell)$, which eliminates the stationary noise that leaks through the sidelobes of the FBF.

Noise Canceller Adaptation

The reference noise signals are emphasized by the ANC and subtracted from the output of the FBF, yielding

$$Y(k,\ell) = \big[\boldsymbol{W}_0^{\mathrm{H}}(k,\ell) - \boldsymbol{G}^{\mathrm{H}}(k,\ell)\mathcal{H}^{\mathrm{H}}(k)\big]\boldsymbol{Z}(k,\ell)\,. \tag{47.24}$$

Let three hypotheses $H_{0\mathrm{s}}$, $H_{0\mathrm{t}}$, and H_1 indicate, respectively, the absence of transients, the presence of an interfering transient, and the presence of a desired source transient at the beamformer output. The optimal solution for the filters $\boldsymbol{G}(k,\ell)$ is obtained by minimizing the power of the beamformer output during the stationary noise frames (i. e., when $H_{0\mathrm{s}}$ is true) [47.2]. We note, however, that no adaptation should be carried out during abrupt changes in the characteristics of the noise signal (e.g., a passing car). When the noise source position is constant and the noise statistics is slowly varying, the ANC filters can track the changes.

Let $\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell) = E\{\boldsymbol{N}_{\mathrm{s}}(k,\ell)\boldsymbol{N}_{\mathrm{s}}^{\mathrm{H}}(k,\ell)\}$ denote the PSD matrix of the input stationary noise. Then, the power of the stationary noise at the beamformer output is minimized by solving the unconstrained optimization problem:

$$\begin{aligned} \min_{\boldsymbol{G}}\{&[\boldsymbol{W}_0(k,\ell) - \mathcal{H}(k,\ell)\boldsymbol{G}(k,\ell)]^{\mathrm{H}} \\ &\times \boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)[\boldsymbol{W}_0(k,\ell) - \mathcal{H}(k,\ell)\boldsymbol{G}(k,\ell)]\}\,. \end{aligned} \tag{47.25}$$

A multichannel Wiener solution is given by (see also [47.42, 46])

$$\begin{aligned} \boldsymbol{G}(k,\ell) = &\big[\mathcal{H}^{\mathrm{H}}(k,\ell)\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)\mathcal{H}(k)\big]^{-1} \\ &\times \mathcal{H}^{\mathrm{H}}(k,\ell)\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)\boldsymbol{W}_0(k,\ell)\,. \end{aligned} \tag{47.26}$$

In practice, this optimization problem is solved by using the normalized LMS algorithm [47.2]:

$$\boldsymbol{G}(k,\ell+1) = \begin{cases} \boldsymbol{G}(k,\ell) + \frac{\mu_g}{P_{\text{est}}(k,\ell)}\boldsymbol{U}(k,\ell)Y^*(k,\ell) & H_{0s} \text{ true}, \\ \boldsymbol{G}(k,\ell)\,, & \text{otherwise}, \end{cases} \tag{47.27}$$

where

$$P_{\text{est}}(k,\ell) = \alpha_p P_{\text{est}}(k,\ell-1) + (1-\alpha_p)\|\boldsymbol{U}(k,\ell)\|^2 \tag{47.28}$$

represents the power of the noise reference signals, μ_g is a step size that regulates the convergence rate, and α_p is a smoothing parameter in the PSD estimation process.

To allow for the use of the STFT, we further assume that the ANC filters $\boldsymbol{g}_m$ have a time-varying finite impulse response (FIR) structure:

$$\boldsymbol{g}_m^{\mathrm{T}}(t) = \left(g_{m,-K_L}(t) \;\; \ldots \;\; g_{m,K_R}(t)\right). \tag{47.29}$$

Note, that the impulse responses are taken to be non-causal, to allow for relative delays between the FBF and the ANC branches.

In order to fulfill the FIR structure constraint in (47.29), the filters update is now given by

$$\tilde{\boldsymbol{G}}(\ell+1,k) = \boldsymbol{G}(k,\ell) + \mu\frac{\boldsymbol{U}(k,\ell)Y^*(k,\ell)}{P_{\text{est}}(k,\ell)}\,,$$
$$\boldsymbol{G}(\ell+1,k) \xleftarrow{\text{FIR}} \tilde{\boldsymbol{G}}(\ell+1,k)\,. \tag{47.30}$$

The operator $\xleftarrow{\text{FIR}}$ includes the following three stages, applied per filter: transformation of $\tilde{G}_m(\ell+1,k)$ to the time domain, truncation of the resulting impulse response to the interval $[-K_L, K_R]$ (i. e., imposing the FIR constraint), and transformation back to the frequency domain. The various filtering operations involved in the algorithm (multiplications in the transform domain) are realized using the overlap-and-save method [47.50, 51].

The resulting algorithm is merely an extension of the original Griffiths and Jim algorithm for the arbitrary ATF case. Figure 47.3 depicts a block diagram of the algorithm. The steps involved in the computation are summarized in Table 47.2. The matched beamformer $\boldsymbol{W}_0(k)$ and the BM $\mathcal{H}(k)$ are assumed to be known at this stage.

47.2.3 Time-Domain Generalized Sidelobe Canceller

The most commonly used GSC structure is the classical time-domain counterpart of the algorithm, proposed

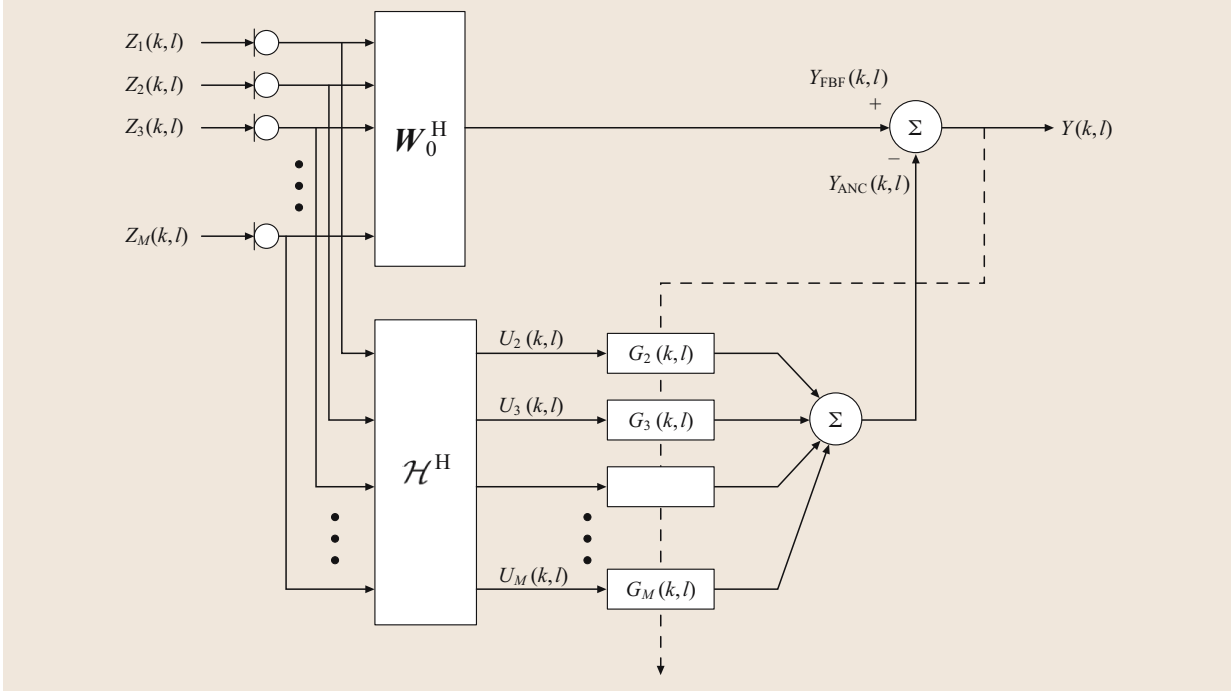


Fig. 47.3 Linearly constrained adaptive beamformer

Table 47.2 Frequency-domain GSC algorithm

1) Fixed beamformer:
$Y_{\mathrm{FBF}}(k,\ell) = \boldsymbol{W}_0^{\mathrm{H}}(k)\boldsymbol{Z}(k,\ell)$
2) Noise reference signals:
$\boldsymbol{U}(k,\ell) = \mathcal{H}^{\mathrm{H}}(k)\boldsymbol{Z}(k,\ell)$
3) Output signal:
$Y(k,\ell) = Y_{\mathrm{FBF}}(k,\ell) - \boldsymbol{G}^{\mathrm{H}}(k,\ell)\boldsymbol{U}(k,\ell)$
4) Filters update
$\tilde{\boldsymbol{G}}(t+1,k) = \boldsymbol{G}(k,\ell) + \mu_{\mathrm{g}} \frac{\boldsymbol{U}(k,\ell)Y^*(k,\ell)}{P_{\mathrm{est}}(k,\ell)}$
$\boldsymbol{G}(\ell+1,k) \xleftarrow{\mathrm{FIR}} \tilde{\boldsymbol{G}}(\ell+1,k)$,
where
$P_{\mathrm{est}}(k,\ell) = \alpha_{\mathrm{p}} P_{\mathrm{est}}(\ell-1,k) + (1-\alpha_{\mathrm{p}}) \sum_m \lvert Z_m(k,\ell)\rvert^2$
5) Keep only nonaliased samples

by *Griffiths* and *Jim* [47.36]. For completeness of the exposition, we present now the time-domain algorithm.

Assuming that the array is steered towards the desired speech signal (refer to steering-related issues in Sect. 47.4), the FBF is given by

$$y_{\mathrm{FBF}}(t) = \sum_{m=1}^{M} z_m(t) \,,$$

which is the simple delay-and-sum beamformer. Under the same delay-only steered array assumptions, it is evident that

$$u_m(t) = z_m(t) - z_1(t);\ m = 2, \ldots, M \,,$$

are noise-only signals and that the desired speech component is cancelled out.

The filters $\boldsymbol{g}_m$ are updated in the time domain. The error signal (which is also the output of the enhancement algorithm) is given by,

$$y(t) = y_{\mathrm{FBF}}(t) - \sum_{m=2}^{M} \sum_{i=-K_L}^{K_R} g_{m,i}(t) u_m(t-i) \,. \tag{47.31}$$

Define, for $m = 2, \ldots, M$:

$$\begin{aligned} &\boldsymbol{u}_m^{\mathrm{T}}(t) \\ &= \big(u_m(t+K_L) \ \cdots \ u_m(t) \ \cdots \ u_m(t-K_R)\big) \,. \end{aligned}$$

Then, the adaptive normalized multichannel LMS solution is given by

$$\begin{aligned} &\boldsymbol{g}_m(t+1) \\ &= \boldsymbol{g}_m(t) + \frac{\mu}{p_{\mathrm{est}}(t)} \boldsymbol{u}_m(t) y(t);\ m = 2, \ldots, M \,, \end{aligned}$$

where

$$p_{\mathrm{est}}(t) = \sum_{m=1}^{M} \lVert \boldsymbol{u}_m(t) \rVert^2 \,. \tag{47.32}$$

47.3 Fixed Beamformer and Blocking Matrix

In the previous section, we derived the generalized sidelobe canceller and showed that it includes a fixed beamformer, given in (47.20), a blocking matrix, given in (47.17), and a multichannel ANC, given in (47.26). Note that knowledge of the ATFs $\boldsymbol{A}(k)$ (assumed to be slowly time variant) suffices to determine both the FBF and BM. In this section we present three methods for determination of the fixed beamformer and the blocking matrix.

47.3.1 Using Acoustical Transfer Functions

A typical RIR is depicted in Fig. 47.4. It can be seen that the impulse response can get very long (several thousand taps), which makes the estimation task quite cumbersome. This impulse response was generated by the image method [47.52] proposed by Allen and Berkley (The authors thank E. A. P. Habets from TU Eindhoven, The Netherlands, for providing an efficient implementation

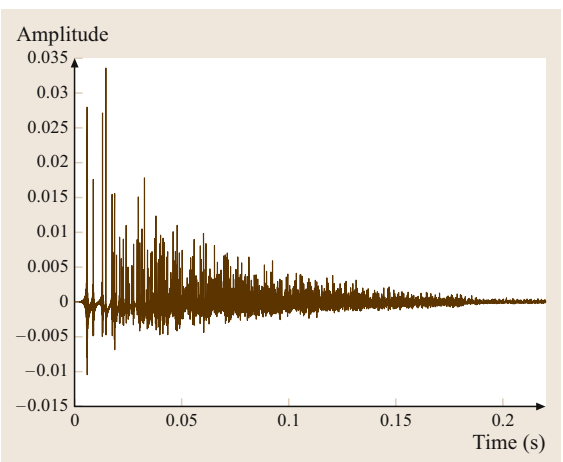


Fig. 47.4 A typical room impulse response with a reverberation time of 0.4 s

of the image method.). By this method, the RIR is generated by simulating multiple reflections of the sound source from the room walls. Two distinct segments of the impulse response can be observed. The first consists of the direct propagation path with a few early, distinguishable reflections. The second segment consists of overlapping random arrivals, with an exponentially decaying envelope. This segment is usually referred to as the tail of the RIR. Using this model we can estimate the various blocks of the GSC.

Assuming that $\boldsymbol{A}(k)$ is known, we have by (47.22), (47.20), and (47.3)

$$Y_{\mathrm{FBF}}(k,\ell) = S(k,\ell) + \frac{\boldsymbol{A}^{\mathrm{H}}(k)}{\|\boldsymbol{A}(k)\|^2}[\boldsymbol{N}_{\mathrm{s}}(k,\ell)+\boldsymbol{N}_{\mathrm{t}}(k,\ell)]\,. \quad (47.33)$$

The first term on the right-hand side is the signal term, and the second is the noise term. The FBF in this case is hence a matched filter-and-sum beamformer (see also [47.16, 17]).

Considering the blocking matrix, there are many alternatives for blocking the desired speech signal in the reference channels. One alternative is calculation of

$$U_m(k,\ell) = A_m(k)Z_{m-1}(k,\ell) - A_{m-1}(k)Z_m(k,\ell)$$

for $m = 2,\ldots,M$. Any other combination of the microphone signals is applicable.

47.3.2 Using Delay-Only Filters

The simplest and yet the most widely used model for the ATF is a delay-only model. Arbitrarily defining the first microphone as the reference microphone we have

$$\boldsymbol{A}(k) = \left(1 \;\; \mathrm{e}^{-\mathrm{i}\frac{2\pi}{K}\tau_2} \;\; \mathrm{e}^{-\mathrm{i}\frac{2\pi}{K}\tau_3} \;\; \ldots \;\; \mathrm{e}^{-\mathrm{i}\frac{2\pi}{K}\tau_M}\right),$$

where $\tau_2,\ldots,\tau_M$ are the relative delays between each microphone and the reference microphone.

In the delay-only case, the FBF simplifies to the delay-and-sum beamformer, given by

$$\boldsymbol{W}_0(k) = \left(1 \;\; \mathrm{e}^{\mathrm{i}\frac{2\pi}{K}\tau_2} \;\; \mathrm{e}^{\mathrm{i}\frac{2\pi}{K}\tau_3} \;\; \ldots \;\; \mathrm{e}^{\mathrm{i}\frac{2\pi}{K}\tau_M}\right).$$

To avoid noncausal delays, a fixed amount of delay can be introduced.

It can easily be verified that the matrix

$$\mathcal{H}(k) = \begin{pmatrix} -\mathrm{e}^{\mathrm{i}\frac{2\pi}{K}\tau_2} & -\mathrm{e}^{\mathrm{i}\frac{2\pi}{K}\tau_2} & \ldots & -\mathrm{e}^{\mathrm{i}\frac{2\pi}{K}\tau_2} \\ 1 & 0 & \ldots & 0 \\ 0 & 1 & \ldots & 0 \\ & \ldots & \ddots & \\ 0 & 0 & \ldots & 1 \end{pmatrix} \quad (47.34)$$

is a proper BM under the assumption of delay-only impulse response. This BM, originally proposed Griffiths and *Jim* [47.36], performs delay compensation and subtraction. It can be regarded as steering $M-1$ null beams towards the desired speech signal.

47.3.3 Using Relative Transfer Functions

We have shown, on the one hand, that the RIR can be very long and hence difficult to estimate. On the other hand, the use of delay-only model suffers from severe undermodeling problems. A good compromise is the use of the relative transfer function (RTF) between sensors. Define the RTFs as the ratio

$$\tilde{\boldsymbol{A}}^{\mathrm{T}}(k) \triangleq \left(1 \;\; \frac{A_2(k)}{A_1(k)} \;\; \frac{A_3(k)}{A_1(k)} \;\; \ldots \;\; \frac{A_M(k)}{A_1(k)}\right) = \frac{\boldsymbol{A}^{\mathrm{T}}(k)}{A_1(k)}\,. \quad (47.35)$$

Note that the ATF may have zeros outside the unit circle, as it is not necessarily a minimum-phase system. Thus to ensure stability of the RTFs we allow for noncausal systems. Therefore, we model the impulse response of the m-th ratio as

$$\tilde{\boldsymbol{a}}_m^{\mathrm{T}}(t) = \left(\tilde{a}_{m,-q_L}(t) \;\; \ldots \;\; \tilde{a}_{m,q_R}(t)\right). \quad (47.36)$$

It was experimentally shown that RTFs are usually much shorter than the corresponding ATFs [47.38], hence the FIR assumption may be justified.

Replacing in (47.20) the actual ATFs by the RTFs, the FBF becomes

$$\boldsymbol{W}_0(k) = \frac{\tilde{\boldsymbol{A}}(k)}{\|\tilde{\boldsymbol{A}}(k)\|^2}\,. \quad (47.37)$$

By (47.22) and (47.3) we then have

$$Y_{\mathrm{FBF}}(k,\ell) = A_1(k)S(k,\ell) + \frac{\tilde{\boldsymbol{A}}^{\mathrm{H}}(k)}{\|\tilde{\boldsymbol{A}}(k)\|^2}[\boldsymbol{N}_{\mathrm{s}}(k,\ell)+\boldsymbol{N}_{\mathrm{t}}(k,\ell)]\,. \quad (47.38)$$

Thus, when $\boldsymbol{W}_0(k,\ell)$ is given by (47.37), the signal term of $Y_{\mathrm{FBF}}(k,\ell)$ is the desired signal distorted only by the first ATF, $A_1(k)$. Note, however, that all the sensor outputs are summed coherently.

It can be easily verified that the use of the following BM suffices for completely eliminating the desired speech signal, provided that the RTFs are correctly modeled and estimated:

$$\mathcal{H}(k)=\begin{pmatrix} -\tilde{A}_2^*(k) & -\tilde{A}_3^*(k) & \dots & -\tilde{A}_M^*(k) \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ & \dots & \ddots & \\ 0 & 0 & \dots & 1 \end{pmatrix} . \quad (47.39)$$

For this choice of the BM, the reference signals are given by,

$$U_m(k,\ell) = Z_m(k,\ell) - \tilde{A}_m(k) Z_1(k,\ell); \quad m = 2, 3, \dots, M . \quad (47.40)$$

47.4 Identification of the Acoustical Transfer Function

The specific choice of the ATFs model governs the applicable estimation method.

47.4.1 Signal Subspace Method

Affes and *Grenier* [47.37] prove that the identification of source-to-array impulse responses is possible by subspace tracking. Assume that the approximation in (47.3) is valid, i. e., the multichannel correlation matrix of the received signals is given by

$$\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell) = \phi_{\mathrm{ss}}(k,\ell)\boldsymbol{A}(k)\boldsymbol{A}^{\mathrm{H}}(k) + \boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell) .$$

Assume also that the noise signal is spatially white, i. e., $\boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell) = \sigma_n^2(k,\ell)\mathbf{I}$ (an extension for spatially nonwhite noise is described in [47.53]). The eigenvalues of the received signals correlation matrix are given by

$$\begin{aligned} &\lambda_l = \sigma_n^2(k,\ell) \quad l = 1, \dots, M-1 \\ &\lambda_M > \sigma_n^2(k,\ell) \ \text{ otherwise} . \end{aligned} \quad (47.41)$$

From the matrix structure we conclude that the most dominant eigenvector of $\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)$ is given by $\boldsymbol{A}(k)$ (up to a scale factor). Hence, using the eigenvalue decomposition of $\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)$, we can estimate the desired signal ATFs. In the case of spatially nonwhite noise signals the generalized eigenvalue decomposition (GEVD), using the known noise correlation matrix $\boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell)$, can be applied instead [47.53].

Yang [47.54] proved that finding the most dominant eigenvectors is equivalent to minimizing a quadratic cost function. For the following derivation it is more convenient to deal with normalized terms. Let

$$\begin{aligned} \boldsymbol{Z}(k,\ell) &= \boldsymbol{A}(k)S(k,\ell) + \boldsymbol{N}(k,\ell) \\ &= \frac{\boldsymbol{A}(k)}{\|\boldsymbol{A}(k)\|}\|\boldsymbol{A}(k)\|S(k,\ell) + \boldsymbol{N}(k,\ell) \\ &\triangleq \bar{\boldsymbol{A}}(k)\bar{S}(k,\ell) + \boldsymbol{N}(k,\ell) , \end{aligned} \quad (47.42)$$

where $\bar{\boldsymbol{A}}(k) \triangleq \boldsymbol{A}(k)/\|\boldsymbol{A}(k)\|$ is the normalized ATFs vector, namely $\bar{\boldsymbol{A}}^{\mathrm{H}}(k)\bar{\boldsymbol{A}}(k) = 1$, and $\bar{S}(k,\ell) \triangleq \|\boldsymbol{A}(k)\| S(k,\ell)$ is the normalized desired speech signal.

When only the single most dominant eigenvector is required (as in the discussed case), *Affes* and *Grenier* [47.37] showed that Yang's criterion simplifies to the following minimization,

$$E\{\|[\mathbf{I} - \bar{\boldsymbol{A}}(k)\bar{\boldsymbol{A}}^{\mathrm{H}}(k)]\boldsymbol{Z}(k,\ell)\|^2\} . \quad (47.43)$$

Similar to the approximation used in the derivation of LMS algorithm, the received signal correlation matrix is approximated by its instantaneous value $\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell) \approx \boldsymbol{Z}(k,\ell)\boldsymbol{Z}^{\mathrm{H}}(k,\ell)$. Furthermore, approximating $\hat{\bar{\boldsymbol{A}}}^{\mathrm{H}}(k,\ell)\hat{\bar{\boldsymbol{A}}}(k,\ell) \approx 1$, where $\hat{\bar{\boldsymbol{A}}}(k,\ell)$ is an estimate of $\bar{\boldsymbol{A}}(k)$ at the current time instant, the following sequential procedure can be derived:

$$\begin{aligned} &\hat{\bar{\boldsymbol{A}}}(k,\ell+1) \\ &= \hat{\bar{\boldsymbol{A}}}(k,\ell) + \mu_{\mathrm{a}}(k,\ell)[\boldsymbol{Z}(k,\ell) \\ &\quad - \hat{\bar{\boldsymbol{A}}}(k,\ell)\hat{\bar{\boldsymbol{A}}}^{\mathrm{H}}(k,\ell)\boldsymbol{Z}(k,\ell)][\hat{\bar{\boldsymbol{A}}}^{\mathrm{H}}(k,\ell)\boldsymbol{Z}(k,\ell)]^* . \end{aligned} \quad (47.44)$$

Define $\bar{Y}_{\mathrm{FBF}}(k,\ell) \triangleq \bar{\boldsymbol{A}}^{\mathrm{H}}(k,\ell)\boldsymbol{Z}(k,\ell)$, and observe that the desired signal component at $\bar{Y}_{\mathrm{FBF}}(k,\ell)$ is $\bar{S}(k,\ell)$. Then,

$$\begin{aligned} &\hat{\bar{\boldsymbol{A}}}(k,\ell+1) \\ &= \hat{\bar{\boldsymbol{A}}}(k,\ell) + \mu_{\mathrm{a}}(k,\ell)[\boldsymbol{Z}(k,\ell) \\ &\quad - \hat{\bar{\boldsymbol{A}}}(k,\ell)\bar{Y}_{\mathrm{FBF}}(k,\ell)]\bar{Y}^*_{\mathrm{FBF}}(k,\ell) . \end{aligned} \quad (47.45)$$

Now, it is easy to verify that

$$\boldsymbol{Z}(k,\ell) - \hat{\bar{\boldsymbol{A}}}(k,\ell)\bar{Y}_{\mathrm{FBF}}(k,\ell)$$

are noise-only signals, provided that the estimate $\hat{\bar{\boldsymbol{A}}}(k,\ell)$ converges to the true normalized ATFs vector $\bar{\boldsymbol{A}}(k)$.

Hence, it can serve as a BM output, namely $\boldsymbol{U}(k,\ell)$. (There is a slight difference between the proposed BM and the conventional BM, as the number of the components in $\boldsymbol{U}(k,\ell)$ in the current scheme is M rather than $M-1$ as with the latter.) Collecting all terms we finally have

$$\hat{\tilde{\boldsymbol{A}}}(k,\ell+1) = \hat{\tilde{\boldsymbol{A}}}(k,\ell)+\mu_{\mathrm{a}}(k,\ell)\boldsymbol{U}(k,\ell)\bar{Y}^{*}_{\mathrm{FBF}}(k,\ell)\,. \tag{47.46}$$

Note that this procedure yields an estimate of the normalized ATFs rather than the ATFs themselves. *Affes* and *Grenier* [47.37] argue that $\|\boldsymbol{A}(k)\|$ is invariant to small talker movements and could be estimated in advance.

As a final remark, we would like to point out that the estimation procedure in (47.46) has many similarities to the method of *Hoshuyama* et al. [47.55] for robust design of the BM, and with the decorrelation criterion presented by *Weinstein* et al. [47.56] and further adapted to the GSC structure by *Gannot* [47.57]. We will elaborate on this issue in Sect. 47.5 when discussing the robust beamformers.

47.4.2 Time Difference of Arrival

When a delay-only steering is applied, an estimation of the time difference of arrival between the microphones suffices to model the entire impulse response. It should be noted however, that this procedure usually undermodels the RIR and is not sufficient for the problem at hand. Many algorithms were proposed for estimation the time difference of arrival (TDOA) [47.58, 59]. A survey of state-of-the-art methods for TDOA estimation can be found in [47.60]. This topic is beyond the scope of this chapter.

47.4.3 Relative Transfer Function Estimation

We present two methods for RTF estimation. The first is based on speech nonstationarity [47.38] , and the second employs the speech presence probability [47.61, 62].

Using Signal Nonstationarity

In this section, we review the system identification technique proposed by *Shalvi* and *Weinstein* [47.63] and later used by *Gannot* et al. [47.38] in the context of microphone arrays. This method relies on the assumptions that the background noise signal is stationary, that the desired signal $s(t)$ is nonstationary, and that the support of the relative impulse response between the sensors is finite and slowly time varying. (Note that the relative impulse response between the sensors is generally of infinite length, since it represents the ratio of ATFs. However, in real environments, the energy of the relative impulse response often decays much faster than the corresponding ATF [47.38]. Therefore, the finite-support assumption is practically not very restrictive.)

Rearranging terms in (47.40) we have

$$Z_m(k,\ell) = \tilde{A}_m(k)Z_1(k,\ell)+U_m(k,\ell)\,. \tag{47.47}$$

We assume that the RTFs are slowly changing in time compared to the time variations of the desired signal. We further assume that the statistics of the noise signal is slowly changing compared to the statistics of the desired signal. Consider some analysis interval during which the ATFs are assumed to be time invariant and the noise signal is assumed to be stationary. We divide that analysis interval into frames. Consider the i-th frame. By (47.47) we have

$$\boldsymbol{\Phi}^{(i)}_{z_m z_1}(k) = \tilde{A}_m(k)\boldsymbol{\Phi}^{(i)}_{z_1 z_1}(k)+\boldsymbol{\Phi}_{u_m z_1}(k)\,, \quad i=1,\ldots,I\,, \tag{47.48}$$

where I is the number of frames, $\boldsymbol{\Phi}^{(i)}_{z_i z_j}(k)$ is the cross-PSD between z_i and z_j at frequency bin k during the i-th frame, and $\boldsymbol{\Phi}_{u_m z_1}(k)$ is the cross-PSD between u_m and z_1 at frequency bin k, which is independent of the frame index due to the noise stationarity. Now, equations (47.2) and (47.40) imply that, when the signal is present,

$$U_m(k,\ell) = N^{\mathrm{s}}_m(k,\ell)-\tilde{A}_m(k)N^{\mathrm{s}}_1(k,\ell) \tag{47.49}$$

$$Z_1(k,\ell) = A_1(k)S(k,\ell)+N^{\mathrm{s}}_1(k,\ell)\,. \tag{47.50}$$

Let $\hat{\boldsymbol{\Phi}}^{(i)}_{z_1 z_1}(k)$, $\hat{\boldsymbol{\Phi}}^{(i)}_{z_m z_1}(k)$, and $\hat{\boldsymbol{\Phi}}^{(i)}_{u_m z_1}(k)$ be estimates of $\boldsymbol{\Phi}^{(i)}_{z_1 z_1}(k)$, $\boldsymbol{\Phi}^{(i)}_{z_m z_1}(k)$, and $\boldsymbol{\Phi}_{u_m z_1}(k)$, respectively. The estimates are obtained by replacing expectations with averages. Note that (47.48) also holds for the estimated values. Let $\varepsilon^{(i)}_m(k) = \hat{\boldsymbol{\Phi}}^{(i)}_{u_m z_1}(k)-\boldsymbol{\Phi}_{u_m z_1}(k)$ denote the estimation error of the cross-PSD between z_1 and u_m in the i-th frame. We then obtain,

$$\hat{\boldsymbol{\Phi}}^{(i)}_{z_m z_1}(k) = \tilde{A}_m(k)\hat{\boldsymbol{\Phi}}^{(i)}_{z_1 z_1}(k)+\boldsymbol{\Phi}_{u_m z_1}(k) +\varepsilon^{(i)}_m(k), \qquad i=1,\ldots,I\,. \tag{47.51}$$

If the noise reference signals $U_m(k,\ell)$, $m=2,\ldots,M$ were uncorrelated with $Z_1(k,\ell)$, then the standard system identification estimate, $\tilde{A}_m(k)=\hat{\boldsymbol{\Phi}}_{z_m z_1}(k)/\hat{\boldsymbol{\Phi}}_{z_1 z_1}(k)$, could be used to obtain an unbiased estimate of $\tilde{A}_m(k)$. Unfortunately, by (47.49) and (47.50), $U_m(k,\ell)$ and $Z_1(k,\ell)$ are in general correlated. Hence in [47.63] it is

proposed to obtain an unbiased estimate of $\tilde{A}_m(k)$ by applying the least-squares (LS) procedure to the following set of overdetermined equations

$$\begin{aligned}\boldsymbol{x} &\triangleq \begin{pmatrix} \hat{\boldsymbol{\Phi}}^{(1)}_{z_m z_1}(k) \\ \hat{\boldsymbol{\Phi}}^{(2)}_{z_m z_1}(k) \\ \vdots \\ \hat{\boldsymbol{\Phi}}^{(K)}_{z_m z_1}(k) \end{pmatrix} \\ &= \begin{pmatrix} \hat{\boldsymbol{\Phi}}^{(1)}_{z_1 z_1}(k) & 1 \\ \hat{\boldsymbol{\Phi}}^{(2)}_{z_1 z_1}(k) & 1 \\ \vdots & \\ \hat{\boldsymbol{\Phi}}^{(K)}_{z_1 z_1}(k) & 1 \end{pmatrix} \begin{pmatrix} \tilde{A}_m(k) \\ \boldsymbol{\Phi}_{u_m z_1}(k) \end{pmatrix} + \begin{pmatrix} \varepsilon^{(1)}_m(k) \\ \varepsilon^{(2)}_m(k) \\ \vdots \\ \varepsilon^{(K)}_m(k) \end{pmatrix} \\ &\triangleq \boldsymbol{G}\,\boldsymbol{\theta} + \boldsymbol{\epsilon}\,, \end{aligned} \tag{47.52}$$

where a separate set of equations is used for each $m = 2, \ldots, M$. The weighted least-squares (WLS) estimate of $\boldsymbol{\theta}$ is obtained by

$$\begin{aligned}\begin{pmatrix} \hat{\tilde{A}}_m(k) \\ \hat{\boldsymbol{\Phi}}_{u_m z_1}(k) \end{pmatrix} &= \hat{\boldsymbol{\theta}} \\ &= \arg\min_{\boldsymbol{\theta}} (\boldsymbol{x} - \boldsymbol{G}\,\boldsymbol{\theta})^{\mathrm{H}} \boldsymbol{W} (\boldsymbol{x} - \boldsymbol{G}\,\boldsymbol{\theta}) \\ &= (\boldsymbol{G}^{\mathrm{H}} \boldsymbol{W} \boldsymbol{G})^{-1} \boldsymbol{G}^{\mathrm{H}} \boldsymbol{W} \boldsymbol{x}\,, \end{aligned} \tag{47.53}$$

where $\boldsymbol{W}$ is a positive Hermitian weighting matrix, and $\boldsymbol{G}^{\mathrm{H}}\boldsymbol{W}\boldsymbol{G}$ is required to be invertible.

Shalvi and Weinstein suggested two alternative weighting matrices. One alternative is given by

$$W_{ij} = \begin{cases} T_i\,, & i = j \\ 0\,, & i \neq j \end{cases}\,, \tag{47.54}$$

where T_i is the length of subinterval i, so that longer intervals have higher weights. In this case, (47.53) reduces to

$$\begin{aligned}&\hat{\tilde{A}}(k) \\ &= \frac{\langle \hat{\phi}_{z_m z_1}(k)\, \hat{\phi}_{z_1 z_1}(k) \rangle - \langle \hat{\phi}_{z_m z_1}(k) \rangle \langle \hat{\phi}_{z_1 z_1}(k) \rangle}{\langle \hat{\phi}^2_{z_1 z_1}(k) \rangle - \langle \hat{\phi}_{z_1 z_1}(k) \rangle^2} \end{aligned} \tag{47.55}$$

with the average operation defined by

$$\langle \varphi(k) \rangle \triangleq \frac{\sum_{i=1}^{I} T_i \varphi^{(i)}(k)}{\sum_{i=1}^{I} T_i}\,. \tag{47.56}$$

Another alternative for $\boldsymbol{W}$, that minimizes the covariance of $\hat{\boldsymbol{\theta}}$, is given by

$$W_{ij} = \frac{B}{\hat{\phi}_{u_m u_m}(k)} \begin{cases} T_i / \hat{\phi}^{(i)}_{z_1 z_1}(k)\,, & i = j \\ 0\,, & i \neq j \end{cases}\,, \tag{47.57}$$

where B is related to the bandwidth of the window used for the cross-PSD estimation [47.63]. With this choice of the weighting function, (47.53) yields

$$\begin{aligned}\hat{\tilde{A}}(k) = &\frac{\langle 1/\hat{\phi}_{z_1 z_1}(k) \rangle \langle \hat{\phi}_{z_m z_1}(k) \rangle}{\langle \hat{\phi}_{z_1 z_1}(k) \rangle \langle 1/\hat{\phi}_{z_1 z_1}(k) \rangle - 1} \\ &- \frac{\langle \hat{\phi}_{z_m z_1}(k)\,/\,\hat{\phi}_{z_1 z_1}(k) \rangle}{\langle \hat{\phi}_{z_1 z_1}(k) \rangle \langle 1/\hat{\phi}_{z_1 z_1}(k) \rangle - 1} \end{aligned} \tag{47.58}$$

and the variance of $\hat{\tilde{A}}(k)$ is given by

$$\mathrm{var}\{\hat{\tilde{A}}(k)\} = \frac{1}{BT} \frac{\phi_{u_m u_m}(k) \langle 1/\phi_{z_1 z_1}(k) \rangle}{\langle \phi_{z_1 z_1}(k) \rangle \langle 1/\phi_{z_1 z_1}(k) \rangle - 1}\,, \tag{47.59}$$

where $T \triangleq \sum_{i=1}^{I} T_i$ is the total observation interval. Special attention should be given to choosing the frame length. On the one hand, it should be longer than the correlation length of $z_m(t)$, which must be longer than the length of the filter $a_m(t)$. On the other hand, it should be short enough for the filter time invariance and the noise quasistationarity assumptions to hold.

A major limitation of the WLS optimization in (47.53) is that both the identification of $\tilde{\boldsymbol{A}}(k)$ and the estimation of the cross-PSD $\phi_{u_m z_1}(k)$ are carried out using the same weight matrix $\boldsymbol{W}$. That is, each subinterval i is given the same weight, whether we are trying to find an estimate for $\tilde{\boldsymbol{A}}(k)$ or for $\phi_{u_m z_1}(k)$. However, subintervals with higher SNR values are of greater importance when estimating $\tilde{A}(k)$, whereas the opposite is true when estimating $\phi_{u_m z_1}(k)$. Consequently, the optimization criterion in (47.53) consists of two conflicting requirements: one is minimizing the error variance of $\tilde{\boldsymbol{A}}(k)$, which pulls the weight up to higher values on higher SNR subintervals. The other requirement is minimizing the error variance of $\phi_{u_m z_1}(k)$, which rather implies *smaller* weights on higher SNR subintervals. For instance, suppose we obtain observations on a relatively long low-SNR interval of length T_0, and on a relatively short high-SNR interval of length T_1 ($T_1 \ll T_0$). Then, the variance of $\tilde{\boldsymbol{A}}(k)$ in (47.59) is inversely proportional to the relative length of the high-SNR interval, $T_1/(T_0 + T_1)$. That is, including in the observation interval additional segments that do not contain speech (i. e., increasing T_0) increases the variance of $\tilde{\boldsymbol{A}}(k)$. This unnatural

consequence is a result of the desire to minimize the variance of $\phi_{u_m z_1}(k)$ by using larger weights on the segments that do not contain speech, while increasing the weights on such subintervals degrades the estimate for $\tilde{A}(k)$.

Another major limitation of RTF identification using nonstationarity is that the interfering signals are required to be stationary during the entire observation interval. The observation interval should include a certain number of subintervals that contain the desired signal, such that $\phi_{z_1 z_1}(k)$ is sufficiently nonstationary for all k. Unfortunately, if the desired signal is speech, the presence of the desired signal in the observed signals may be sparse in some frequency bands. This entails a very long observation interval, thus constraining the interfering signals to be stationary over long intervals. Furthermore, the RTF $\tilde{\boldsymbol{A}}(k)$ is assumed to be constant during the observation interval. Hence, very long observation intervals also restrict the capability of the system identification technique to track varying $\tilde{\boldsymbol{A}}(k)$ (e.g., tracking moving talkers in reverberant environments).

Using Speech Presence Probability

In this section, we present a system identification approach that is adapted to speech signals. Specifically, the presence of the desired speech signal in the time–frequency domain is uncertain, and the speech presence probability is utilized to separate the tasks of system identification and cross-PSD estimation. An estimate for $\tilde{\boldsymbol{A}}(k)$ is derived based on subintervals that contain speech, while subintervals that do not contain speech are of more significance when estimating the components of $\phi_{u_m z_1}(k)$.

Let the observed signals be divided in time into overlapping frames by the application of a window function and analyzed using the STFT. Under the same considerations leading to the estimation procedure based on speech nonstationarity (and based on the assumption that the RTFs can be modelled by short filters), (47.48) is still valid. Now using (47.48)-(47.50) and the fact that the desired signal $s(t)$ is uncorrelated with the interfering signals $n_m^{\mathrm{s}}(t)$; $m = 1, 2, \ldots, M$, we have

$$\begin{aligned}\phi_{z_m z_1}(k,\ell) &= \tilde{A}_m(k)|A_1|^2(k)\phi_{\mathrm{s\,s}}(k,\ell) \\ &\quad + \phi_{n_m^{\mathrm{s}} n_1^{\mathrm{s}}}(k,\ell)\,. \end{aligned} \tag{47.60}$$

Writing this equation in terms of the PSD estimates, we obtain for $m = 2, 3, \ldots, M$

$$\begin{aligned}\hat{\phi}_{z_m z_1}(k,\ell) &= \tilde{A}_m(k)|\hat{A}_1|^2(k)\hat{\phi}_{\mathrm{s\,s}}(k,\ell) \\ &\quad + \hat{\phi}_{n_m^{\mathrm{s}} n_1^{\mathrm{s}}}(k,\ell) + \varepsilon_m(k,\ell) \\ &= \tilde{A}_m(k)\hat{\phi}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}(k,\ell) + \hat{\phi}_{n_m^{\mathrm{s}} n_1^{\mathrm{s}}}(k,\ell) + \varepsilon_m(k,\ell)\,, \end{aligned} \tag{47.61}$$

where $\varepsilon_m(k,\ell)$ denotes an estimation error and $\hat{\phi}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}(k,\ell) = |\hat{A}_1|^2(k)\hat{\phi}_{\mathrm{s\,s}}(k,\ell)$ represents the PSD of the speech signal component in microphone 1. This gives us L equations, which may be written in a matrix form as

$$\begin{aligned}\hat{\boldsymbol{\psi}}_m(k) &\triangleq \begin{pmatrix} \hat{\phi}_{z_m z_1}(k,1) - \hat{\phi}_{n_m^{\mathrm{s}} n_1^{\mathrm{s}}}(k,1) \\ \hat{\phi}_{z_m z_1}(k,2) - \hat{\phi}_{n_m^{\mathrm{s}} n_1^{\mathrm{s}}}(k,2) \\ \vdots \\ \hat{\phi}_{z_m z_1}(k,L) - \hat{\phi}_{n_m^{\mathrm{s}} n_1^{\mathrm{s}}}(k,L) \end{pmatrix} \\ &= \begin{pmatrix} \hat{\phi}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}(k,1) \\ \hat{\phi}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}(k,2) \\ \vdots \\ \hat{\phi}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}(k,L) \end{pmatrix} \tilde{A}_m(k) + \begin{pmatrix} \varepsilon_m(k,1) \\ \varepsilon_m(k,2) \\ \vdots \\ \varepsilon_m(k,L) \end{pmatrix} \\ &\triangleq \hat{\boldsymbol{\phi}}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}(k)\,\tilde{A}_m(k) + \boldsymbol{\varepsilon}_m(k)\,. \end{aligned} \tag{47.62}$$

Since the RTF $\tilde{\boldsymbol{A}}_m(k)$ represents the coupling between the primary and reference sensors with respect to the *desired* source signal, the optimization criterion for the identification of $\tilde{A}_m(k)$ has to take into account only short-time frames which contain desired signal components. Specifically, let $I(k,\ell)$ denote an indicator function for the signal presence [$I(k,\ell) = 1$ if $\phi_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}(k,\ell) \neq 0$, i. e., during H_1, and $I(k,\ell) = 0$ otherwise], and let $\mathbf{I}(k)$ represent a diagonal matrix with the elements $[I(k,1), I(k,2), \ldots, I(k,L)]$ on its diagonal. Then the WLS estimate of $\tilde{A}(k)$ is obtained by

$$\begin{aligned}\hat{\tilde{A}}_m &= \arg\min_{\tilde{A}_m} \left\{ [\mathbf{I}\boldsymbol{\varepsilon}_m]^{\mathrm{H}} \mathbf{W} [\mathbf{I}\boldsymbol{\varepsilon}_m] \right\} \\ &= \arg\min_{\tilde{A}_m} \Big\{ [\hat{\boldsymbol{\psi}}_m - \hat{\boldsymbol{\phi}}_{\tilde{\mathrm{s}}\tilde{\mathrm{s}}} \tilde{A}_m]^{\mathrm{H}} \\ &\quad \mathbf{I}\mathbf{W}\mathbf{I} [\hat{\boldsymbol{\psi}}_m - \hat{\boldsymbol{\phi}}_{\tilde{\mathrm{s}}\tilde{\mathrm{s}}} \tilde{A}_m] \Big\} \\ &= [\hat{\boldsymbol{\phi}}_{\tilde{\mathrm{s}}\tilde{\mathrm{s}}}^{\mathrm{T}} \mathbf{I}\mathbf{W}\mathbf{I} \hat{\boldsymbol{\phi}}_{\tilde{\mathrm{s}}\tilde{\mathrm{s}}}]^{-1} \hat{\boldsymbol{\phi}}_{\tilde{\mathrm{s}}\tilde{\mathrm{s}}}^{\mathrm{T}} \mathbf{I}\mathbf{W}\mathbf{I} \hat{\boldsymbol{\psi}}_m\,, \end{aligned} \tag{47.63}$$

where the argument k has been omitted for notational simplicity. Recognizing the product $\mathbf{I}\,\mathbf{W}\,\mathbf{I}$ as the equivalent weight matrix, the variance of $\hat{\tilde{A}}$ is given by ([47.64] p. 405)

$$\begin{aligned}\mathrm{var}\{\hat{\tilde{A}}_m\} &= \left(\boldsymbol{\phi}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}^{\mathrm{T}} \mathbf{I}\mathbf{W}\mathbf{I} \boldsymbol{\phi}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}\right)^{-1} \boldsymbol{\phi}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}^{\mathrm{T}} \mathbf{I}\mathbf{W}\mathbf{I}\,\mathrm{cov}(\boldsymbol{\varepsilon}_m) \\ &\quad \times \mathbf{I}\mathbf{W}\mathbf{I} \boldsymbol{\phi}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}} \left(\boldsymbol{\phi}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}^{\mathrm{T}} \mathbf{I}\mathbf{W}\mathbf{I} \boldsymbol{\phi}_{\tilde{\mathrm{s}}\,\tilde{\mathrm{s}}}\right)^{-1}\,, \end{aligned} \tag{47.64}$$

where $\mathrm{cov}(\boldsymbol{\varepsilon}_m)$ is the covariance matrix of $\boldsymbol{\varepsilon}_m$. The matrix $\mathbf{W}$ that minimizes the variance of $\hat{\tilde{A}}$ therefore satisfies ([47.64] prop. 8.2.4)

$$\mathbf{I}\mathbf{W}\mathbf{I} = \mathbf{I}\,[\mathrm{cov}(\boldsymbol{\varepsilon}_m)]^{-1}\,\mathbf{I}\,. \tag{47.65}$$

This choice of $\mathbf{W}$ yields an asymptotically unbiased estimator

$$\hat{A} = \left\{\hat{\boldsymbol{\phi}}_{\tilde{s}\tilde{s}}{}^{\mathrm{T}}\mathbf{I}[\mathrm{cov}(\boldsymbol{\varepsilon}_m)]^{-1}\mathbf{I}\hat{\boldsymbol{\phi}}_{\tilde{s}\tilde{s}}\right\}^{-1} \times \hat{\boldsymbol{\phi}}_{\tilde{s}\tilde{s}}{}^{\mathrm{T}}\mathbf{I}[\mathrm{cov}(\boldsymbol{\varepsilon}_m)]^{-1}\mathbf{I}\hat{\boldsymbol{\psi}}_m \,, \tag{47.66}$$

which is known as the *minimum variance* or Gauss–Markov estimator. Substituting (47.65) into (47.64), we obtain the variance of the resulting estimator

$$\mathrm{var}\{\hat{A}\} = \left\{\boldsymbol{\phi}_{\tilde{s}\tilde{s}}{}^{\mathrm{T}}\mathbf{I}[\mathrm{cov}(\boldsymbol{\varepsilon}_m)]^{-1}\mathbf{I}\,\boldsymbol{\phi}_{\tilde{s}\tilde{s}}\right\}^{-1} . \tag{47.67}$$

The elements of $\mathrm{cov}(\boldsymbol{\varepsilon}_m)$ are asymptotically given by (see [47.61])

$$\mathrm{cov}[\varepsilon_m(k,\ell), \varepsilon_m(k,\ell')] = \begin{cases} \phi_{\tilde{s}\tilde{s}}(k,\ell)\,\phi_{u_m u_m}(k,\ell) \,, & \text{if } \ell = \ell' \,, \\ 0 \,, & \text{otherwise.} \end{cases} \tag{47.68}$$

Under the assumption that hypothesis $H_{0\mathrm{t}}$ is false, i. e., the noise is stationary, $\phi_{u_m u_m}(k,\ell)$ is independent of the frame index ℓ (in practice, it suffices that the statistics of the interfering signals is slowly changing compared with the statistics of the desired signal). Denoting by $\langle\cdot\rangle_\ell$ an average operation over the frame index ℓ

$$\langle\varphi(k,\ell)\rangle_\ell \triangleq \frac{1}{L}\sum_{\ell=1}^{L}\varphi(k,\ell) \,, \tag{47.69}$$

and substituting (47.68) into (47.66) and (47.67) we obtain

$$\hat{\tilde{A}}_m(k) = \frac{\langle I(k,\ell)[\hat{\phi}_{z_m z_1}(k,\ell) - \hat{\phi}_{n_m^{\mathrm{s}} n_1^{\mathrm{s}}}(k,\ell)]\rangle_\ell}{\langle I(k,\ell)\,\hat{\phi}_{\tilde{s}\tilde{s}}(k,\ell)\rangle_\ell} \,, \tag{47.70}$$

$$\mathrm{var}\{\hat{\tilde{A}}_m(k)\}\frac{\phi_{u_m u_m}(k)}{L\,\langle I(k,\ell)\,\phi_{\tilde{s}\tilde{s}}(k,\ell)\rangle_\ell} \,. \tag{47.71}$$

Note that, for a given frequency-bin index k, only frames that contain speech $[I(k,\ell) \neq 0]$ influence the values of $\hat{\tilde{A}}_m(k)$ and $\mathrm{var}\{\hat{\tilde{A}}_m(k)\}$. In contrast to the nonstationarity method, including in the observation interval additional segments that do not contain speech does not increase the variance of $\hat{\tilde{A}}_m(k)$ for any k. However, the proposed identification approach requires an estimate for $I(k,\ell)$, i. e. identifying which time–frequency bins (k,ℓ) contain the desired signal. In practice, the speech presence probability $p(k,\ell)$ can be estimated from the beamformer output (see Sect. 47.6), and an estimate for the indicator function is obtained by

$$\hat{I}(k,\ell) = \begin{cases} 1 \,, & \text{if } p(k,\ell) \geq p_0 \,, \\ 0 \,, & \text{otherwise,} \end{cases} \tag{47.72}$$

where p_0 $(0 \leq p_0 < 1)$ is a predetermined threshold. Note, also that the output of the beamformer consists of the filtered version of the desired speech $\tilde{s}(t)$, when the RTFs is used in the FBF branch. The parameter p_0 controls the trade-off between the detection and false alarm probabilities, which are defined by $P_{\mathrm{D}} \triangleq \mathcal{P}\{p(k,\ell) \geq p_0 \,|\, I(k,\ell) = 1\}$ and $P_{\mathrm{FA}} \triangleq \mathcal{P}\{p(k,\ell) \geq p_0 \,|\, I(k,\ell) = 0\}$. A smaller value of p_0 increases the detection probability and allows for more short-time frames to be involved in the estimation of $\tilde{\boldsymbol{A}}(k)$. However, a smaller value of p_0 also increases the false alarm probability, which may cause a mismodification of $\hat{\tilde{\boldsymbol{A}}}(k)$ due to frames that do not contain desired speech components.

The identification algorithm based on speech presence probability requires estimates for $\phi_{z_m z_1}(k,\ell)$, $\phi_{\tilde{s}\tilde{s}}(k,\ell)$, and $\phi_{n_m^{\mathrm{s}} n_1^{\mathrm{s}}}(k,\ell)$. An estimate for $\phi_{z_m z_1}(k,\ell)$ is obtained by applying a first-order recursive smoothing to the cross-periodogram of the observed signals, $Z_m(k,\ell)\,Z_1^*(k,\ell)$. Specifically,

$$\hat{\phi}_{z_m z_1}(k,\ell) = \alpha_{\mathrm{s}}\,\hat{\phi}_{z_m z_1}(k,\ell-1) + (1-\alpha_{\mathrm{s}})Z_m(k,\ell)\,Z_1^*(k,\ell) \,, \tag{47.73}$$

where the smoothing parameter α_{s} $(0 \leq \alpha_{\mathrm{s}} < 1)$ determines the equivalent number of cross-periodograms that are averaged, $N_\ell \approx (1+\alpha_{\mathrm{s}})/(1-\alpha_{\mathrm{s}})$. Typically, speech periodograms are recursively smoothed with an equivalent rectangular window of $T_{\mathrm{s}} = 0.2$ s long, which represents a good compromise between smoothing the noise and tracking the speech spectral variations [47.65]. Therefore, for a sampling rate of 8 kHz, an STFT window length of 256 samples and a frame update step of 128 samples, we use $\alpha_{\mathrm{s}} = (T_{\mathrm{s}} \cdot 8000/128 - 1)/(T_{\mathrm{s}} \cdot 8000/128 + 1) \approx 0.85$.

To obtain an estimate for the PSD of the desired signal, we can use the output of the multichannel postfilter discussed in Sect. 47.6, during speech presence, i. e., H_1 is true.

$$\hat{\phi}_{\tilde{s}\tilde{s}}(k,\ell) = \alpha_{\mathrm{s}}\,\hat{\phi}_{\tilde{s}\tilde{s}}(k,\ell-1) + (1-\alpha_{\mathrm{s}})I(k,\ell)\,|Y(k,\ell)|^2 \,. \tag{47.74}$$

The cross-PSD of the interfering signals, $n_m^{\mathrm{s}}(t)$ and $n_1^{\mathrm{s}}(t)$, is estimated by using the minima-controlled

recursive averaging (MCRA) approach [47.66, 67]. Specifically, past spectral cross-power values of the noisy observed signals are recursively averaged with a time-varying frequency-dependent smoothing parameter during periods for which H_{0s} is true

$$\hat{\phi}_{n_m^s n_1^s}(k,\ell) = \tilde{\alpha}_u(k,\ell)\,\hat{\phi}_{n_m^s n_1^s}(k,\ell-1) + \beta\,[1-\tilde{\alpha}_u(k,\ell)]\,Z_m(k,\ell)\,Z_1^*(k,\ell)\,, \tag{47.75}$$

where $\tilde{\alpha}_u(k,\ell)$ is the smoothing parameter ($0 < \tilde{\alpha}_u(k,\ell) \le 1$), and β ($\beta \ge 1$) is a factor that compensates the bias when the desired signal is absent [47.67]. The smoothing parameter is determined by the signal presence probability, $p(k,\ell)$, and a constant α_u ($0 < \alpha_u < 1$) that represents its minimal value

$$\tilde{\alpha}_u(k,\ell) = \alpha_u + (1-\alpha_u)p(k,\ell)\,. \tag{47.76}$$

The value of $\tilde{\alpha}_u$ is close to 1 when the desired signal is present to prevent the noise cross-PSD estimate from increasing as a result of signal components. It decreases linearly with the probability of signal presence to allow a faster update of the noise estimate. The value of α_u compromises between the tracking rate (response rate to abrupt changes in the noise statistics) and the variance of the noise estimate. Typically, in the case of high levels of nonstationary noise, a good compromise is obtained by $\alpha_u = 0.85$ [47.67]. Substituting these spectral estimates into (47.70) we obtain an estimate for $\tilde{A}_m(k)$.

47.5 Robustness and Distortion Weighting

Beamformers often suffer from sensitivity to signal mismatch. The GSC in particular suffers from two basic problems. First, nonideal FBF can lead to noncoherent filter-and-sum operation. *Doclo* and *Moonen* [47.27] and *Nordholm* et al. [47.68] use spatial and frequency-domain constraints to improve the robustness of beamformers. The second problem, which is the concern of this survey, is the leakage phenomenon, caused by imperfect BM. If the desired speech leaks into the noise reference signals $\boldsymbol{U}(k,\ell)$ the noise canceller filters will subtract speech components from the FBF output, causing self-cancellation of the desired speech, and hence a severe distortion. Note that, even when the ANC filters are adapted during noise-only periods, the self-cancellation is unavoidable. The goal of this section is to present several concepts for increasing the robustness of the GSC structure and reducing its sensitivity to signal mismatch.

Cox et al. [47.7] presented a thorough analysis of array sensitivity. The array output SNR is evidently given by

$$\mathrm{SNR}_{\mathrm{out}}(k,\ell) = \frac{\phi_{ss}(k,\ell)\boldsymbol{W}^{\mathrm{H}}(k)\boldsymbol{A}(k)\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{W}(k)}{\boldsymbol{W}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell)\boldsymbol{W}(k)}\,.$$

Now assume that the signal's ATFs are different from the ATFs used for designing the LCMV beamformer, i. e., $\breve{\boldsymbol{A}}(k) = \boldsymbol{A} + \boldsymbol{\epsilon}(k)$. Assume also that the spatial correlation matrix of the perturbation $\boldsymbol{\epsilon}(k)$ is given by $E\{\boldsymbol{\epsilon}(k)\boldsymbol{\epsilon}^{\mathrm{H}}(k)\} = \sigma_\epsilon^2\mathbf{I}$, where $\mathbf{I}$ is the identity matrix of dimensions $M \times M$. Namely, we assume that the perturbation components are uncorrelated. Hence the expected output SNR is given by

$$E\{\mathrm{SNR}_{\mathrm{out}}(k,\ell)\} = \frac{\phi_{ss}(k,\ell)\boldsymbol{W}^{\mathrm{H}}(k)\big(\boldsymbol{A}(k)\boldsymbol{A}^{\mathrm{H}}(k) + \sigma_\epsilon^2\mathbf{I}\big)\boldsymbol{W}(k)}{\boldsymbol{W}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell)\boldsymbol{W}(k)}\,. \tag{47.77}$$

Define the sensitivity of the array to the ATFs perturbation as $J(k,\ell)$

$$J(k,\ell) \triangleq \frac{\frac{\partial}{\partial\sigma_\epsilon^2}E\{\mathrm{SNR}_{\mathrm{out}}(k,\ell)\}}{E\{\mathrm{SNR}_{\mathrm{out}}(k,\ell)\}|_{\sigma_\epsilon^2=0}} = \frac{\boldsymbol{W}^{\mathrm{H}}(k)\boldsymbol{W}(k)}{\boldsymbol{W}^{\mathrm{H}}(k)\boldsymbol{A}(k)\boldsymbol{A}^{\mathrm{H}}(k)\boldsymbol{W}(k)}\,. \tag{47.78}$$

The resulting expression is the reciprocal of the white-noise gain of the array. Using the array constraint $\boldsymbol{W}^{\mathrm{H}}(k)\boldsymbol{A}(k) = 1$ we finally obtain the following expression for the sensitivity of the array to the ATFs perturbation,

$$J(k,\ell) = \boldsymbol{W}^{\mathrm{H}}(k)\boldsymbol{W}(k)\,. \tag{47.79}$$

Specifically, the array sensitivity is equal to the norm of the beamformer weights. Hence, reducing the sensitivity of the array is equivalent to constraining the norm of the array filter coefficients. Due to (47.15) the array filters can be decomposed into two orthogonal filters, $\boldsymbol{W}(k) = \boldsymbol{W}_0(k) - \boldsymbol{V}(k) = \boldsymbol{W}_0(k) - \mathcal{H}(k)\boldsymbol{G}(k,\ell)$. It is therefore sufficient to constrain the adaptive filter norm, namely $\boldsymbol{G}^{\mathrm{H}}(k,\ell)\boldsymbol{G}(k,\ell) = \|\boldsymbol{G}(k,\ell)\|^2 \le \Omega(k,\ell)$, where $\Omega(k,\ell)$ is a prespecified norm. The GSC structure is

modified to fulfil the norm constraint, as follows:

$$\tilde{\boldsymbol{G}}'(k) = \boldsymbol{G}(k,\ell) + \mu \frac{\boldsymbol{U}(k,\ell)Y^*(k,\ell)}{P_{\text{est}}(k,\ell)}\,, \tag{47.80}$$

$$\tilde{\boldsymbol{G}}(\ell+1,k) = \begin{cases} \tilde{\boldsymbol{G}}'(k) & \|\tilde{\boldsymbol{G}}'(k)\|^2 \le \Omega(k,\ell+1) \\ \frac{\sqrt{\Omega(k,\ell+1)}}{\|\tilde{\boldsymbol{G}}'(k)\|}\tilde{\boldsymbol{G}}'(k) & \text{otherwise.} \end{cases} \tag{47.81}$$

Finally, the conventional FIR constraint is imposed on the norm-constrained filters

$$\boldsymbol{G}(\ell+1,k) \xleftarrow{\text{FIR}} \tilde{\boldsymbol{G}}(\ell+1,k)\,.$$

Based on this concept, *Hoshuyama* et al. [47.55, 69] proposed several methods for addressing the robustness issue, concentrating on the self-cancellation phenomenon, caused by the leakage of the desired speech signal to the BM outputs $\boldsymbol{U}(k,\ell)$. This phenomenon is emphasized in reverberant environments, in the case where the BM only compensates for the relative delay [as in (47.34)]. In general there are two ways to mitigate this leakage problem. First, an improved spatial filtering can be incorporated into the design of the BM. *Claesson* and *Nordholm* [47.22] proposed to apply spatial high-pass filter to cancel out all signals within a specified frequency and angular range. *Huarng* and *Yeh* [47.45] analyzed the leakage phenomenon and applied a derivative constraint on the array response, yielding wider tolerance to pointing errors.

A second cure for the leakage problems involves applying constraints on the ANC filters. *Hoshuyama* et al. [47.55] proposed several structures combining modifications for both the BM and the ANC blocks. The conventional delay-compensation BM is replaced by an adaptive BM based on signal cancellers. Two constraining strategies may be applied to the involved filters. The first strategy uses norm-constraint, and the second uses the leaky LMS adaptation scheme. *Haykin* [47.1] proved that both strategies are equivalent. The modified BM outputs, for $m = 1, \ldots, M$, are given by

$$U_m(k,\ell) = Z_m(k,\ell) - H_m^*(k,\ell)Y_{\text{FBF}}(k,\ell)\,, \tag{47.82}$$

where $H_m(k,\ell)$ are updated as to minimize the power of $U_m(k,\ell)$, by cancelling all desired speech components. Whenever, the SNR in $Y_{\text{FBF}}(k,\ell)$ is sufficiently high, the blocking ability of the structure is improved.

Gannot [47.57] showed that this equation, in conjunction with the expression for the beamformer output

$$Y(k,\ell) = Y_{\text{FBF}}(k,\ell) - \boldsymbol{G}^{\text{H}}(k,\ell)\boldsymbol{U}(k,\ell)\,,$$

is closely related to the decorrelation criterion proposed by *Weinstein* et al. [47.56]. A different decorrelation based structure was later proposed by *Fancourt* and *Parra* [47.70].

Two alternative schemes are proposed for adapting the filters $H_m(k,\ell)$. (Originally, *Hoshuyama* et al. [47.55] stated their formulation in the time domain using the original GSC structure. Here we state the frequency-domain counterpart of the proposed algorithm. The first to propose frequency domain implementation of Hoshuyama's concepts were *Herbordt* and *Kellermann* [47.71, 72].) The first scheme is the leaky LMS,

$$H_m(k,\ell+1) = (1-\delta)H_m(k,\ell) + \frac{\mu_{\text{h}}}{|Y_{\text{FBF}}(k,\ell)|^2}U_m(k,\ell)Y_{\text{FBF}}^*(k,\ell) \tag{47.83}$$

for $m = 1, \ldots, M$. The regular FIR constraint, omitted for the clarity of the exposition, is then applied. The second scheme constrains the filter coefficients to a predefined mask, yielding for $m = 1, \ldots, M$:

$$H'_m(k,\ell+1) = H_m(k,\ell) + \frac{\mu_{\text{h}}}{|Y_{\text{FBF}}(k,\ell)|^2}U_m(k,\ell)Y_{\text{FBF}}^*(k,\ell) \tag{47.84}$$

and

$$H(\ell+1,k) = \begin{cases} \phi_{\text{low}}(k,\ell+1) & H'_m(k,\ell+1) \ge \phi_{\text{low}}(k,\ell+1)\,, \\ \phi_{\text{up}}(k,\ell+1) & H'_m(k,\ell+1) \le \phi_{\text{up}}(k,\ell+1)\,, \\ H'_m(k,\ell+1) & \text{otherwise.} \end{cases} \tag{47.85}$$

The ANC filter is either adapted by the leaky LMS algorithm or the norm-constrained adaptation mechanism proposed by *Cox* (see (47.81)). As a concluding remark summarizing Hoshuyama's methods, we draw the reader attention to the resemblance of the proposed adaptation of the BM filters and the subspace tracking procedure presented by *Affes* and *Grenier* depicted in (47.46).

Spriet et al. [47.33] adopted a different approach to mitigating the leakage problem, by modifying the adaptation criterion for the ANC filters, $\boldsymbol{G}(k,\ell)$. The minimization criterion in (47.25) is altered to deal with the leakage problem. Let, $Y_{\text{FBF}}^{\text{s}}(k,\ell)$ be the speech component at the FBF output. Let $\boldsymbol{U}^{\text{s}}(k,\ell)$ and $\boldsymbol{U}^{\text{n}}(k,\ell)$ be the speech and noise (without distinction between stationary and transient noise signals) components in the

reference signals, respectively. Then, the filters $\boldsymbol{G}(k,\ell)$ minimize the following expression

$$E\{\|Y^{\mathrm{s}}_{\mathrm{FBF}}(k,\ell)-\boldsymbol{G}^{\mathrm{H}}(k,\ell)(\boldsymbol{U}^{\mathrm{s}}(k,\ell)+\boldsymbol{U}^{\mathrm{n}}(k,\ell))\|^2\}\,.$$

Since the speech and noise signals are uncorrelated, the above expression can be restated as

$$E\{\|\boldsymbol{G}^{\mathrm{H}}(k,\ell)\boldsymbol{U}^{\mathrm{n}}(k,\ell)\|^2\} + E\{\|Y^{\mathrm{s}}_{\mathrm{FBF}}(k,\ell)-\boldsymbol{G}^{\mathrm{H}}(k,\ell)\boldsymbol{U}^{\mathrm{s}}(k,\ell)\|^2\}\,. \quad (47.86)$$

Note, that the first term is related to the noise signal and the second term to the speech distortion. Hence, the Wiener filter design criterion can be easily generalized [47.26] to allow for a trade-off between speech distortion and NR, by incorporating a weighting factor $\mu\in[0,\infty)$. The resulting criterion is then given by

$$\mu E\{\|\boldsymbol{G}^{\mathrm{H}}(k,\ell)\boldsymbol{U}^{\mathrm{n}}(k,\ell)\|^2\} + E\{\|Y^{\mathrm{s}}_{\mathrm{FBF}}(k,\ell)-\boldsymbol{G}^{\mathrm{H}}(k,\ell)\boldsymbol{U}^{\mathrm{s}}(k,\ell)\|^2\}\,. \quad (47.87)$$

It is easily verified that the corresponding minimizer is

$$\boldsymbol{G}(k,\ell)=\left(\frac{1}{\mu}\boldsymbol{\Phi}_{\boldsymbol{U}^{\mathrm{s}}\boldsymbol{U}^{\mathrm{s}}}+\boldsymbol{\Phi}_{\boldsymbol{U}^{\mathrm{n}}\boldsymbol{U}^{\mathrm{n}}}\right)^{-1}\boldsymbol{\phi}_{\boldsymbol{U}Y^{\mathrm{s}}_{\mathrm{FBF}}}\,, \quad (47.88)$$

where

$$\boldsymbol{\Phi}_{\boldsymbol{U}^{\mathrm{s}}\boldsymbol{U}^{\mathrm{s}}}=E\{\boldsymbol{U}^{\mathrm{s}}(k,\ell)(\boldsymbol{U}^{\mathrm{s}}(k,\ell))^{\mathrm{H}}\}\,,$$
$$\boldsymbol{\Phi}_{\boldsymbol{U}^{\mathrm{n}}\boldsymbol{U}^{\mathrm{n}}}=E\{\boldsymbol{U}^{\mathrm{n}}(k,\ell)(\boldsymbol{U}^{\mathrm{n}}(k,\ell))^{\mathrm{H}}\}\,,$$
$$\boldsymbol{\phi}_{\boldsymbol{U}Y^{\mathrm{s}}_{\mathrm{FBF}}}=E\{\boldsymbol{U}^{\mathrm{s}}(k,\ell)(Y^{\mathrm{s}}_{\mathrm{FBF}}(k,\ell))^{*}\}\,.$$

Using the reference signals definitions we have

$$\boldsymbol{\Phi}_{\boldsymbol{U}^{\mathrm{s}}\boldsymbol{U}^{\mathrm{s}}}=\mathcal{H}^{\mathrm{H}}(k,\ell)\boldsymbol{\Phi}_{\boldsymbol{SS}}(k,\ell)\mathcal{H}(k,\ell)\,,$$
$$\boldsymbol{\Phi}_{\boldsymbol{U}^{\mathrm{n}}\boldsymbol{U}^{\mathrm{n}}}=\mathcal{H}^{\mathrm{H}}(k,\ell)\boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell)\mathcal{H}(k,\ell)\,,$$
$$\boldsymbol{\phi}_{\boldsymbol{U}^{\mathrm{s}}Y^{\mathrm{s}}_{\mathrm{FBF}}}=\mathcal{H}^{\mathrm{H}}(k,\ell)\boldsymbol{\Phi}_{\boldsymbol{SS}}(k,\ell)\boldsymbol{W}_0(k,\ell)\,. \quad (47.89)$$

Since $\boldsymbol{\Phi}_{\boldsymbol{SS}}(k,\ell)$ is not available it can be evaluated using

$$\boldsymbol{\Phi}_{\boldsymbol{SS}}(k,\ell)=\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)-\boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell)$$

and $\boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell)$ is estimated while the speech signal is absent. *Doclo* and *Moonen* [47.73] prove that the output SNR after NR with the above speech distortion weighted multichannel Wiener filter (SDW-MWF) is always larger than or equal to the input SNR, for any filter length, and for any value of the trade-off parameter μ between NR and speech distortion.

This solution for the ANC filters constitutes the speech distortion regularized generalized sidelobe canceller (SDR-GSC) structure. *Spriet* et al. [47.33] further proposed to incorporate a single-channel postfilter, which compensates for the distortion imposed by the structure in case of speech leakage into the reference signals. Further discussion of this structure is beyond the scope of this survey. In [47.74], the authors propose a stochastic gradient-based implementation of their criterion. The robustness of both the multichannel Wiener filter and the GSC structures are analyzed by *Spriet* et al. [47.75] in the context of hearing-aid application.

Improving the robustness of the BM is an ongoing research topic. An interesting direction was taken by *Low* et al. [47.76]. The authors propose to incorporate concepts adopted from the BSS discipline to improve the separation of the speech and noise signals and hence reducing the amount of leakage of the desired signal into the reference noise signals.

47.6 Multichannel Postfiltering

Postfiltering methods for multimicrophone speech enhancement algorithms have recently attracted an increased interest. It is well known that beamforming methods yield a significant improvement in speech quality [47.9]. However, when the noise field is spatially incoherent or diffuse, the NR is insufficient [47.77] and additional postfiltering is normally required [47.78]. Furthermore, as nonstationary noise cannot, in general, be distinguished from speech signals, a significant performance degradation is expected in nonstationary noise environment.

Most multimicrophone speech enhancement methods consist of a multichannel part (either delay and sum beamformer or GSC [47.36]) followed by a postfilter, which is based on Wiener filtering (sometimes in conjunction with spectral subtraction). Numerous articles have been published on the subject, e.g., [47.79–87] to mention just a few.

In general, the postfilters can be divided into two groups. The first is a single-channel postfilter which acts as a single-microphone speech enhancement algorithm on the beamformer output. Multichannel postfilters, on the other hand, explicitly use the spatial information, extracted by the GSC structure, to gain better distinction between the speech signal and the transient noise.

47.6.1 MMSE Postfiltering

Simmer et al. [47.78] address the general problem of the single-channel postfilter. They first derive the multichannel Wiener filter for estimating the speech signal, $S(k,\ell)$ from the microphone signals $\boldsymbol{Z}(k,\ell)$ given in (47.3). Then, they show that the Wiener filter can be factorized into a multiplication of the LCMV (Frost) beamformer given in (47.7) and a single-channel Wiener filter that depends on the output speech and noise signals.

To show this, we will start with the multichannel Wiener filter, which is given by

$$\boldsymbol{W}^{\text{Wiener}}(k,\ell) = \boldsymbol{\Phi}_{\boldsymbol{ZZ}}^{-1}(k,\ell)\boldsymbol{\Phi}_{\boldsymbol{ZS}}(k,\ell)\,, \tag{47.90}$$

where $\boldsymbol{\Phi}_{\boldsymbol{ZZ}}$ is given in (47.10) and

$$\begin{aligned}\boldsymbol{\Phi}_{\boldsymbol{ZS}}(k,\ell) &= E\{\boldsymbol{Z}(k,\ell)S^*(k,\ell)\}\\ &= \boldsymbol{A}(k)\phi_{\text{ss}}(k,\ell)\,.\end{aligned} \tag{47.91}$$

Hence,

$$\begin{aligned}&\boldsymbol{W}^{\text{Wiener}}(k,\ell)\\ &= \left(\phi_{\text{ss}}(k,\ell)\boldsymbol{A}(k)\boldsymbol{A}^{\text{H}}(k) + \boldsymbol{\Phi}_{\boldsymbol{NN}}(k,\ell)\right)^{-1}\\ &\quad\times\phi_{\text{ss}}(k,\ell)\boldsymbol{A}(k)\,.\end{aligned} \tag{47.92}$$

Omitting, for the clarity of the exposition, the explicit time- and frequency-domain dependence and using the matrix inversion lemma yields

$$\begin{aligned}&\boldsymbol{W}^{\text{Wiener}}(k,\ell)\\ &= \left(\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1} - \frac{\phi_{\text{ss}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}}{1+\phi_{\text{ss}}\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}}\right)\phi_{\text{ss}}\boldsymbol{A}\\ &= \left(1 - \frac{\phi_{\text{ss}}\boldsymbol{A}\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}}{1+\phi_{\text{ss}}\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}}\right)\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\phi_{\text{ss}}\boldsymbol{A}\\ &= \left(\frac{\phi_{\text{ss}}}{1+\phi_{\text{ss}}\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}}\right)\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}\\ &= \left(\frac{\phi_{\text{ss}}}{\phi_{\text{ss}}+(\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A})^{-1}}\right)\frac{\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}}{\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}}\,.\end{aligned} \tag{47.93}$$

(The derivation here is a slight modification of the results introduced in [47.78].) The reader can easily identify the second multiplier as the MSNR beamformer (47.8) which was shown to be equivalent to the LCMV beamformer (47.11).

We turn now to analyzing the first term in the multiplicative expression. The PSD of desired signal component at the output of the LCMV beamformer is given by

$$\begin{aligned}&\phi_{Y_{\text{s}}Y_{\text{s}}}(k,\ell)\\ &= \phi_{\text{ss}}(\boldsymbol{W}^{\text{LCMV}})^{\text{H}}\boldsymbol{A}\boldsymbol{A}^{\text{H}}\boldsymbol{W}^{\text{LCMV}}\\ &= \phi_{\text{ss}}\left(\frac{\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}}{\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}}\right)^2 = \phi_{\text{ss}}\,.\end{aligned} \tag{47.94}$$

As expected, the LCMV is a distortionless beamformer. The noise component at the beamformer output is given by,

$$\begin{aligned}&\phi_{Y_nY_n}(k,\ell)\\ &= (\boldsymbol{W}^{\text{LCMV}})^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}\boldsymbol{W}^{\text{LCMV}} = \frac{\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}}{\left(\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}\right)^2}\\ &= \frac{1}{\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}}\,.\end{aligned} \tag{47.95}$$

Using (47.94) and (47.95), the first term in the multichannel Wiener filter can be rewritten as

$$\begin{aligned}&\frac{\phi_{\text{ss}}}{\phi_{\text{ss}}+\left(\boldsymbol{A}^{\text{H}}\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}\boldsymbol{A}\right)^{-1}}\\ &= \frac{\phi_{Y_{\text{s}}Y_{\text{s}}}(k,\ell)}{\phi_{Y_nY_n}(k,\ell)+\phi_{Y_{\text{s}}Y_{\text{s}}}(k,\ell)}\,,\end{aligned} \tag{47.96}$$

which is evidently recognized as the single-channel Wiener filter applied to the LCMV beamformer output. It was therefore proven that the multichannel Wiener filter can be factorized into a product of the LCMV beamformer and a single-channel Wiener postfilter applied to the beamformer output,

$$\begin{aligned}&\boldsymbol{W}^{\text{Wiener}}(k,\ell)\\ &= \underbrace{\frac{\phi_{Y_{\text{s}}Y_{\text{s}}}(k,\ell)}{\phi_{Y_nY_n}(k,\ell)+\phi_{Y_{\text{s}}Y_{\text{s}}}(k,\ell)}}_{\text{Wiener postfilter}}\\ &\quad\times\underbrace{\frac{\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}(k,\ell)\boldsymbol{A}(k)}{\boldsymbol{A}^{\text{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{NN}}^{-1}(k,\ell)\boldsymbol{A}(k)}}_{\text{LCMV beamformer}}\,.\end{aligned} \tag{47.97}$$

Several algorithms have been proposed for designing the postfilter; all differ in their treatment of the single-channel Wiener postfilter estimation. *Zelinski* [47.79] was probably the first to apply a postfilter to the output of a microphone array (delay and sum beamformer in his formulation). Zelinski proposed the following Wiener

filter estimation

$$\boldsymbol{W}^{\text{Zelinski}}(k,\ell) = \frac{\frac{2}{M(M-1)}\sum_{i=1}^{M-1}\sum_{j=i+1}^{M}\text{Re}[Z_i(k,\ell)Z_j^*(k,\ell)]}{\frac{1}{M}\sum_{i=1}^{M}|Z_i(k,\ell)|^2}\,. \tag{47.98}$$

Later, postfiltering was incorporated into the Griffiths and Jim GSC beamformer by *Bitzer* et al. [47.86, 87]. The authors proposed to use two postfilters in succession. The first is applied to the FBF branch, and the second to the GSC output. In directional noise source and in the low-frequency band of a diffused noise field, correlation between the noise components at each sensor exists. While the first postfilter is useless in this case, the latter suppresses the noise.

Simmer and *Wasiljeff* [47.88] showed that Zelinski's postfilter has two disadvantages. First, only minor SNR improvement can be expected in frequencies, for which the coherence function between the received noise signals is high (i. e., coherent noise sources). Second, for frequencies with a low coherence function, the noise PSD is overestimated by a factor M (the number of microphones). They propose to mitigate the second disadvantage by slightly modifying the noise estimation, to obtain an estimate of the noise PSD at the output of the beamformer, rather than at its input.

In diffused noise field the noise coherence function tends to be high in lower frequencies, whereas in higher-frequency bands it tends to be low. The cutoff frequency depends on the distance between microphones (see further discussion in Sect. 47.7). This property is the cause for the first drawback of Zelinski's postfilter. It was therefore proposed by *Fischer* and *Kammeyer* [47.83] to split the beamformer into three nonoverlapping subarrays (with different inter-microphone distances) for which the noise coherence function is kept low. To avoid grating lobes, bandpass filters with corresponding cutoff frequencies, are applied to the beamformer output. *Marro* et al. [47.43] improved this concept and further modified the Wiener postfilter estimation. A comprehensive survey of these postfiltering methods can be found in [47.78]. *McCowan* and *Bourlard* [47.89, 90] develop a more-general expression of the postfilter estimation, based on an assumed knowledge of the complex coherence function of the noise field. This general expression can be used to construct a more-appropriate postfilter in a variety of different noise fields.

47.6.2 Log-Spectral Amplitude Postfiltering

A major drawback of single-channel postfiltering techniques is that highly nonstationary noise components are not addressed. The time variation of the interfering signals is assumed to be sufficiently slow, such that the postfilter can track and adapt to the changes in the noise statistics. Unfortunately, transient interferences are often much too brief and abrupt for the conventional tracking methods.

Transient Beam-to-Reference Ratio

Generally, the TF-GSC output comprises three components: a nonstationary desired source component, a pseudostationary noise component, and a transient interference. Our objective is to determine which category a given time–frequency bin belongs to, based on the beamformer output and the reference signals.

Recall the three hypotheses H_{0s}, H_{0t}, and H_1 that indicate, respectively, the absence of transients, the presence of an interfering transient, and the presence of a desired source transient at the beamformer output (the pseudostationary interference is present in any case). Then, if transients have not been detected at the beamformer output and the reference signals, we can accept the H_{0s} hypothesis. If a transient is detected at the beamformer output but not at the reference signals, the transient is likely a source component and therefore we determine that H_1 is true. On the contrary, a transient that is detected at one of the reference signals but not at the beamformer output is likely an interfering component, which implies that H_{0t} is true. If a transient is simultaneously detected at the beamformer output and at one of the reference signals, a further test is required, which involves the ratio between the transient power at beamformer output and the transient power at the reference signals. The discussion here is partly based on [47.77]. A real-time version of the method that incorporates adaptive estimation of the ATFs is introduced in [47.91].

Let $\mathcal{S}$ be a smoothing operator in the power-spectral domain,

$$\mathcal{S}Y(k,\ell) = \alpha_s \cdot \mathcal{S}Y(k,\ell-1) + (1-\alpha_s)\sum_{i=-w}^{w} b_i |Y(k-i,\ell)|^2\,, \tag{47.99}$$

where α_s $(0 \le \alpha_s \le 1)$ is a forgetting factor for the smoothing in time, and b is a normalized window function $(\sum_{i=-w}^{w} b_i = 1)$ that determines the order of

smoothing in frequency. Let $\mathcal{M}$ denote an estimator for the PSD of the background pseudostationary noise, derived using the MCRA approach [47.66,67]. The decision rules for detecting transients at the TF-GSC output and reference signals are

$$\Lambda_Y(k,\ell) \triangleq \mathcal{S}Y(k,\ell)/\,\mathcal{M}Y(k,\ell) > \Lambda_0\,, \tag{47.100}$$

$$\Lambda_{\boldsymbol{U}}(k,\ell) \triangleq \max_{2\le i\le M}\left\{\frac{\mathcal{S}U_i(k,\ell)}{\mathcal{M}U_i(k,\ell)}\right\} > \Lambda_1\,, \tag{47.101}$$

respectively, where Λ_Y and Λ_U denote measures of the local nonstationarities, and Λ_0 and Λ_1 are the corresponding threshold values for detecting transients [47.92]. The transient beam-to-reference ratio (TBRR) is defined by the ratio between the transient power of the beamformer output and the transient power of the strongest reference signal

$$\Omega(k,\ell) = \frac{\mathcal{S}Y(k,\ell) - \mathcal{M}Y(k,\ell)}{\max\limits_{2\le i\le M}\{\mathcal{S}U_i(k,\ell) - \mathcal{M}U_i(k,\ell)\}}\,. \tag{47.102}$$

Transient signal components are relatively strong at the beamformer output, whereas transient noise components are relatively strong at one of the reference signals. Hence, we expect $\Omega(k,\ell)$ to be large for signal transients, and small for noise transients. Assuming there exist thresholds $\Omega_{\text{high}}(k)$ and $\Omega_{\text{low}}(k)$ such that

$$\Omega(k,\ell)|_{H_{0t}} \le \Omega_{\text{low}}(k) \le \Omega_{\text{high}}(k) \le \Omega(k,\ell)|_{H_1} \tag{47.103}$$

the decision rule for differentiating desired signal components from the transient interference components is

$$\begin{aligned} &H_{0t}: \gamma_s(k,\ell) \le 1 \text{ or } \Omega(k,\ell) \le \Omega_{\text{low}}(k)\,,\\ &H_1: \gamma_s(k,\ell) \ge \gamma_0 \text{ and } \Omega(k,\ell) \ge \Omega_{\text{high}}(k)\,,\\ &H_r: \text{otherwise}, \end{aligned} \tag{47.104}$$

where

$$\gamma_s(k,\ell) \triangleq \frac{|Y(k,\ell)|^2}{\mathcal{M}Y(k,\ell)} \tag{47.105}$$

represents the a posteriori SNR at the beamformer output with respect to the pseudostationary noise, γ_0 denotes a constant satisfying $\mathcal{P}(\gamma_s(k,\ell) \ge \gamma_0 \mid H_{0s}) < \epsilon$ for a certain significance level ϵ, and H_r designates a *reject* option where the conditional error of making a decision between H_{0t} and H_1 is high.

Figure 47.5 summarizes a block diagram for the hypothesis testing. The hypothesis testing is carried out in the time–frequency plane for each frame and frequency bin. H_{0s} is accepted when transients have neither been detected at the beamformer output nor at the reference signals. If a transient is detected at the beamformer output but not at the reference signals, we accept H_1. On the other hand, if a transient is detected at one of the reference signals but not at the beamformer output, we accept H_{0t}. If a transient is detected simultaneously at the beamformer output and at one of the reference signals, we compute the TBRR $\Omega(k,\ell)$ and the a posteriori SNR at the beamformer output with respect to the pseudostationary noise $\gamma_s(k,\ell)$ and decide on the hypothesis according to (47.104).

Log-Spectral Amplitude Estimation

We address now the problem of estimating the time-varying PSD of the TF-GSC output noise component, and present the multichannel postfiltering technique. Figure 47.6 describes a block diagram of the multichannel postfiltering. Following the hypothesis testing, an estimate $\hat{q}(k,\ell)$ for the a priori signal absence probability is produced. Subsequently, we derive an estimate $p(k,\ell) \triangleq \mathcal{P}(H_1 \mid Y, \boldsymbol{U})$ for the signal presence probability, and an estimate $\hat{\lambda}_d(k,\ell)$ for the noise PSD.

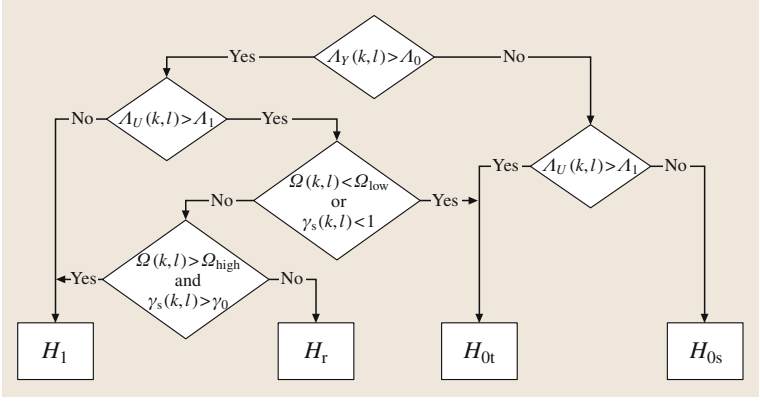


Fig. 47.5 Block diagram for hypothesis testing

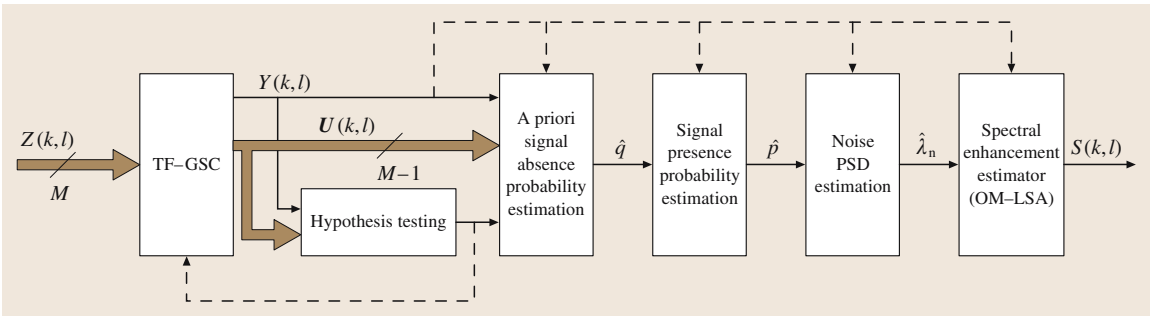


Fig. 47.6 Block diagram of multichannel postfiltering

Finally, spectral enhancement of the beamformer output is achieved by applying the optimally-modified log-spectral amplitude (OM-LSA) gain function [47.93], which minimizes the MSE of the log-spectral amplitude under signal presence uncertainty.

Based on a Gaussian statistical model [47.94], the signal presence probability is given by

$$p(k,\ell) = \left\{1+\frac{q(k,\ell)}{1-q(k,\ell)}[1+\xi(k,\ell)]\exp[-\upsilon(k,\ell)]\right\}^{-1}, \tag{47.106}$$

where $\xi(k,\ell) \triangleq \lambda_s(k,\ell)/\lambda_n(k,\ell)$ is the a priori SNR, $\lambda_s(k,\ell)$ is the desired signal PSD at the beamformer output $\lambda_n(k,\ell)$ is the noise PSD at the beamformer output, $\upsilon(k,\ell) \triangleq \gamma(k,\ell)\,\xi(k,\ell)/[1+\xi(k,\ell)]$, and $\gamma(k,\ell) \triangleq |Y(k,\ell)|^2/\lambda_n(k,\ell)$ is the a posteriori SNR. The a priori signal absence probability $\hat{q}(k,\ell)$ is set to 1 if the signal absence hypotheses (H_{0s} or H_{0t}) are accepted, and is set to 0 if the signal presence hypothesis (H_1) is accepted. In the case of the reject hypothesis H_r, a soft signal detection is accomplished by letting $\hat{q}(k,\ell)$ be inversely proportional to $\Omega(k,\ell)$ and $\gamma_s(k,\ell)$:

$$\hat{q}(k,\ell) = \max\left\{\frac{\gamma_0-\gamma_s(k,\ell)}{\gamma_0-1}, \frac{\Omega_{high}-\Omega(k,\ell)}{\Omega_{high}-\Omega_{low}}\right\}. \tag{47.107}$$

The a priori SNR is estimated by [47.93]

$$\hat{\xi}(k,\ell) = \alpha K_{H_1}^2(k,\ell-1)\gamma(k,\ell-1) + (1-\alpha)\max\{\gamma(k,\ell)-1, 0\}, \tag{47.108}$$

where α is a weighting factor that controls the trade-off between NR and signal distortion, and

$$K_{H_1}(k,\ell) \triangleq \frac{\xi(k,\ell)}{1+\xi(k,\ell)}\exp\left(\frac{1}{2}\int_{\upsilon(k,\ell)}^{\infty}\frac{e^{-t}}{t}\,dt\right) \tag{47.109}$$

is the spectral gain function of the log-spectral amplitude (LSA) estimator when signal is surely present [47.95]. An estimate for noise PSD is obtained by recursively averaging past spectral power values of the noisy measurement, using a time-varying frequency-dependent smoothing parameter. The recursive averaging is given by

$$\hat{\lambda}_n(k,\ell+1) = \tilde{\alpha}_n(k,\ell)\hat{\lambda}_n(k,\ell)+\beta[1-\tilde{\alpha}_n(k,\ell)]|Y(k,\ell)|^2, \tag{47.110}$$

where the smoothing parameter $\tilde{\alpha}_n(k,\ell)$ is determined by the signal presence probability $p(k,\ell)$,

$$\tilde{\alpha}_n(k,\ell) \triangleq \alpha_n+(1-\alpha_n)\,p(k,\ell), \tag{47.111}$$

and β is a factor that compensates the bias when signal is absent. The constant α_n ($0<\alpha_n<1$) represents the minimal smoothing parameter value. The smoothing parameter is close to 1 when signal is present, to prevent an increase in the noise estimate as a result of signal components. It decreases when the probability of signal presence decreases, to allow a fast update of the noise estimate.

Table 47.3 Values of the parameters used in the implementation of the log-spectral amplitude postfiltering for a sampling rate of 8 kHz

Normalized LMS:	$\alpha_p = 0.9$	$\mu_h = 0.05$
ATF identification:	$N = 10$	$R = 10$
Hypothesis testing:	$\alpha_s = 0.9$	$\gamma_0 = 4.6$
	$\Lambda_0 = 1.67$	$\Lambda_1 = 1.81$
	$\Omega_{low} = 1$	$\Omega_{high} = 3$
	$b = (0.25\ \ 0.5\ \ 0.25)$	
Noise PSD estimation:	$\alpha_n = 0.85$	$\beta = 1.47$
Spectral enhancement:	$\alpha = 0.92$	
	$K_{min} = -20\,\text{dB}$	

The estimate of the clean signal STFT is finally given by

$$\hat{S}(k,\ell) = K(k,\ell)Y(k,\ell)\,, \tag{47.112}$$

where

$$K(k,\ell) = \left\{K_{H_1}(k,\ell)\right\}^{p(k,\ell)} K_{\min}^{1-p(k,\ell)} \tag{47.113}$$

is the OM-LSA gain function and $K_{\min}$ denotes a lower bound constraint for the gain when signal is absent. The implementation of the integrated TF-GSC and multichannel postfiltering algorithm is summarized in Fig. 47.6.

Typical values of the respective parameters, for a sampling rate of 8 kHz, are given in Table 47.3. The STFT and its inverse are implemented with biorthogonal Hamming windows of 256 samples length (32 ms) and 64 samples frame update step (75% overlapping windows).

47.7 Performance Analysis

The use of actual signals (such as noisy speech recordings in room environment) demonstrates the ability of the TF-GSC algorithm to reduce the noise while maintaining the desired signal spectral content (Sect. 47.8). However, it is also beneficial to perform analytical evaluation of the expected performance, especially for determining the performance limits. While the D-GSC [47.36] is widely analyzed in the literature, the more-realistic arbitrary ATFs scenario, is only superficially treated. In more-complex environments such as reverberating room this assumption is not valid, and may result in severe degradation in the performance. Furthermore, most references address the NR obtained by the algorithm but do not present any measure of distortion imposed on the desired signal, even in the simple delay-only ATFs. In this section we will analyze both the NR of the TF-GSC structure (while using the RTFs rather than the ATFs themselves) and the distortion it imposes. For a thorough performance evaluation of the multichannel postfilter (for the two-channel case) please refer to [47.96].

47.7.1 The Power Spectral Density of the Beamformer Output

Using (47.24) and (47.37), the algorithm's output is given by

$$Y(k,\ell) = \frac{\hat{\tilde{\boldsymbol{A}}}^{\mathrm{H}}(k)}{\|\hat{\tilde{\boldsymbol{A}}}(k)\|^2}\boldsymbol{Z}(k,\ell) - \boldsymbol{G}^{\mathrm{H}}(k,\ell)\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{Z}(k,\ell)\,,$$

where only estimates of the RTFs, $\hat{\tilde{\boldsymbol{A}}}(k)$ [and $\widehat{\mathcal{H}}(k)$], rather than their exact values, are assumed to be known. Using this expression, the PSD of the output signal is given by

$$\begin{aligned}
\boldsymbol{\Phi}_{\mathrm{yy}}(k,\ell) &= E[Y(k,\ell)Y^*(k,\ell)] \\
&= E\left\{\left[\frac{1}{\|\hat{\tilde{\boldsymbol{A}}}(k)\|^2}\hat{\tilde{\boldsymbol{A}}}^{\mathrm{H}}(k)\boldsymbol{Z}(k,\ell)\right.\right. \\
&\quad \left.- \boldsymbol{G}^{\mathrm{H}}(k,\ell)\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{Z}(k,\ell),\right] \\
&\quad \times\left[\frac{1}{\|\hat{\tilde{\boldsymbol{A}}}(k)\|^2}\hat{\tilde{\boldsymbol{A}}}^{\mathrm{H}}(k)\boldsymbol{Z}(k,\ell)\right. \\
&\quad \left.\left.- \boldsymbol{G}^{\mathrm{H}}(k,\ell)\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{Z}(k,\ell),\right]^{\mathrm{H}}\right\}\,.
\end{aligned} \tag{47.114}$$

Opening brackets and using the PSD definition $\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell) = E\{\boldsymbol{Z}(k,\ell)\boldsymbol{Z}^{\mathrm{H}}(k,\ell)\}$ yields,

$$\begin{aligned}
\boldsymbol{\Phi}_{\mathrm{yy}}(k,\ell) &= \frac{1}{\|\hat{\tilde{\boldsymbol{A}}}(k)\|^4}\hat{\tilde{\boldsymbol{A}}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\hat{\tilde{\boldsymbol{A}}}(k) \\
&\quad - \frac{1}{\|\hat{\tilde{\boldsymbol{A}}}(k)\|^2}\boldsymbol{G}^{\mathrm{H}}(k,\ell)\widehat{\mathcal{H}}^{\mathrm{H}}(k) \\
&\quad \times\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\hat{\tilde{\boldsymbol{A}}}(k) \\
&\quad - \frac{1}{\|\hat{\tilde{\boldsymbol{A}}}(k)\|^2}\hat{\tilde{\boldsymbol{A}}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\widehat{\mathcal{H}}(k)\boldsymbol{G}(k) \\
&\quad + \boldsymbol{G}^{\mathrm{H}}(k,\ell)\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\widehat{\mathcal{H}}(k)\boldsymbol{G}(k)\,.
\end{aligned} \tag{47.115}$$

The output PSD depends on the input signal $\boldsymbol{Z}(k,\ell)$ and the optimal multichannel Wiener filter, given by (47.26), calculated during frames for which hypothesis $H_{0\mathrm{s}}$ is true. Using the independence of the desired signal and the noise signal, the NR and the distortion imposed by the algorithm can be calculated separately by deriving

expressions for the output PSD in the following two situations:

$$\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)=\begin{cases}\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell) & H_{0\mathrm{s}}\,,\\ \phi_{\mathrm{ss}}(k,\ell)\boldsymbol{A}(k)\boldsymbol{A}^{\mathrm{H}}(k) & H_1\,,\end{cases}$$

yielding

$$\boldsymbol{\Phi}_{\mathrm{yy}}(k,\ell)=\begin{cases}\boldsymbol{\Phi}^{\mathrm{n}}_{\mathrm{yy}}(k,\ell) & H_{0\mathrm{s}} \Rightarrow \text{noise reduction},\\ \boldsymbol{\Phi}^{\mathrm{s}}_{\mathrm{yy}}(k,\ell) & H_1 \Rightarrow \text{distortion}\,.\end{cases}$$

Note that for simplicity we calculate the NR only during $H_{0\mathrm{s}}$, i. e., while only stationary noise signal is present. For a performance evaluation in the nonstationary case, please refer to [47.96].

Using (47.26) and (47.115), we obtain the output signal PSD:

$$\begin{aligned}\boldsymbol{\Phi}_{\mathrm{yy}}(k,\ell)=\frac{1}{\|\hat{\tilde{\boldsymbol{A}}}(k)\|^4}\times\Big\{&\hat{\tilde{\boldsymbol{A}}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\hat{\tilde{\boldsymbol{A}}}(k)\\ &-\hat{\tilde{\boldsymbol{A}}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)\widehat{\mathcal{H}}(k)\\ &\quad\left[\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)\widehat{\mathcal{H}}(k)\right]^{-1}\\ &\quad\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\hat{\tilde{\boldsymbol{A}}}(k)\\ &-\hat{\tilde{\boldsymbol{A}}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\widehat{\mathcal{H}}(k)\\ &\quad\left[\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)\widehat{\mathcal{H}}(k)\right]^{-1}\\ &\quad\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)\hat{\tilde{\boldsymbol{A}}}(k)\\ &+\hat{\tilde{\boldsymbol{A}}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)\widehat{\mathcal{H}}(k)\\ &\quad\left[\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)\widehat{\mathcal{H}}(k)\right]^{-1}\\ &\quad\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)\widehat{\mathcal{H}}(k)\\ &\times\left[\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)\widehat{\mathcal{H}}(k)\right]^{-1}\\ &\quad\widehat{\mathcal{H}}^{\mathrm{H}}(k)\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)\hat{\tilde{\boldsymbol{A}}}(k)\Big\}\,.\end{aligned}\tag{47.116}$$

This complicated expression depends on various parameters: the input signal PSD, $\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)$, the stationary noise PSD used for calculating the optimal filters, $\boldsymbol{\Phi}_{\boldsymbol{N}_{\mathrm{s}}\boldsymbol{N}_{\mathrm{s}}}(k,\ell)$, and the RTFs estimate $\hat{\tilde{\boldsymbol{A}}}(k)$ [which is also used for the BM $\widehat{\mathcal{H}}(k)$]. This expression will be used in Subsects. 47.7.2 and 47.7.3 for deriving general expressions for the distortion imposed by the algorithm and the obtainable NR, respectively. These general expressions will be evaluated for several interesting cases.

47.7.2 Signal Distortion

The distortion imposed by the algorithm can be calculated by the general expression given in (47.116) for a signal $\boldsymbol{Z}(k,\ell)=\boldsymbol{A}(k)s(k,\ell)$. Assume a perfect estimate of the RTFs $\tilde{\boldsymbol{A}}(k)$, is available, i. e., $\hat{\tilde{\boldsymbol{A}}}(k)=\tilde{\boldsymbol{A}}(k)=\frac{\boldsymbol{A}(k)}{A_1(k)}$. Thus, using the signal PSD expression $\boldsymbol{\Phi}_{\boldsymbol{ZZ}}(k,\ell)=\phi_{\mathrm{ss}}(k,\ell)\boldsymbol{A}(k)\boldsymbol{A}^{\mathrm{H}}(k)$ and the identity $\mathcal{H}^{\mathrm{H}}(k)\boldsymbol{A}(k)=0$, expression (47.116) reduces to

$$\begin{aligned}&\boldsymbol{\Phi}^{\mathrm{s}}_{\mathrm{yy}}(k,\ell)\\ &=\phi_{\mathrm{ss}}(k,\ell)\frac{|\mathcal{F}(k)|^2}{\|\tilde{\boldsymbol{A}}(k)\|^4}\tilde{\boldsymbol{A}}^{\mathrm{H}}(k)\boldsymbol{A}(k)\boldsymbol{A}^{\mathrm{H}}(k)\tilde{\boldsymbol{A}}(k)\\ &\quad\phi_{\mathrm{ss}}(k,\ell)|A_1(k)|^2\,.\end{aligned}\tag{47.117}$$

The filter $A_1(k)$ is the ATF relating the source signal and the first (arbitrarily chosen as the reference) sensor. This distortion cannot be eliminated by the algorithm. Note that this distortion is due to the use of the RTFs, rather than the ATFs themselves. Using direct estimate of the ATFs will avoid this distortion affect. Actually, it imposes on the output signal the same amount of distortion imposed on the arbitrary reference sensor. Hence, we define the total distortion caused by the algorithm by normalizing the output,

$$\mathrm{DIS}(k,\ell)=\frac{\boldsymbol{\Phi}^{\mathrm{s}}_{\mathrm{yy}}(k,\ell)}{|A_1(k)|^2\phi_{\mathrm{ss}}(k,\ell)}\,.\tag{47.118}$$

Hence, a value of $\mathrm{DIS}(k,\ell)=1$ indicates a distortionless output. This value is obtained whenever an exact knowledge of the RTFs is available.

The distortion level demonstrates only weak dependence on the noise field, both for the delay-only and complex ATF cases. For details please refer to [47.77].

47.7.3 Stationary Noise Reduction

We calculate now the amount of obtainable stationary NR. When the noise is nonstationary, the use of a postfilter becomes more important. A performance analysis of the multichannel postfilter (for the two-channel case) in nonstationary noise environment can be found in [47.96].

We will use again the general expression for the output signal given by (47.116), this time with a noise signal as the input signal, i. e., $\boldsymbol{Z}(k,\ell)=\boldsymbol{N}_{\mathrm{s}}(k,\ell)$ [the same noise signal used for calculating the optimal Wiener filter (47.26)]. The expression (47.116) now reduces to

$$
\begin{aligned}
&\Phi_{yy}^{n}(k,\ell)\\
&=\frac{\hat{\tilde{A}}^{H}(k)\Phi_{N_sN_s}(k,\ell)\hat{\tilde{A}}(k)}{\|\hat{\tilde{A}}(k)\|^4}-\frac{\hat{\tilde{A}}^{H}(k)}{\|\hat{\tilde{A}}(k)\|^4}\\
&\quad\times\Phi_{N_sN_s}(k,\ell)\widehat{\mathcal{H}}(k)\left[\widehat{\mathcal{H}}^{H}(k)\Phi_{N_sN_s}(k,\ell)\right]^{-1}\\
&\quad\times\widehat{\mathcal{H}}^{H}(k)\Phi_{N_sN_s}(k,\ell)\hat{\tilde{A}}(k)\,. \qquad (47.119)
\end{aligned}
$$

The first term of the equation can be identified as $\Phi_{FBF}^{n}(k,\ell)$, the FBF component of the output PSD when a noise-only signal is applied to the array,

$$
\begin{aligned}
\Phi_{FBF}^{n}(k,\ell)&=E\left\{Y_{FBF}^{n}(k,\ell)\left[Y_{FBF}^{n}(k,\ell)\right]^{*}\right\}\\
&=\frac{\hat{\tilde{A}}^{H}(k)\Phi_{N_sN_s}(k,\ell)\hat{\tilde{A}}(k)}{\|\hat{\tilde{A}}(k)\|^4}\,. \qquad (47.120)
\end{aligned}
$$

Another interesting figure of merit is the extra NR obtained by the noise cancelling branch (see also [47.40]),

$$
\mathrm{NR}_{ANC}(k,\ell)=\frac{\Phi_{FBF}^{n}(k,\ell)}{\Phi_{yy}^{n}(k,\ell)}\,. \qquad (47.121)
$$

Expressions (47.119), (47.120), and (47.121) can be used for calculating the NR obtained by the algorithm and to determine the major contributor for this NR. Assuming small errors regime, the error in estimating $\tilde{A}(k)$ has only minor influence on the NR. Therefore, we will assume, throughout the NR analysis, perfect knowledge of the RTFs, i. e., $\hat{\tilde{A}}(k)=\tilde{A}(k)$.

The resulting expressions for the noise cancellation depends on the noise PSD at the sensors. We calculate now the expected NR of the algorithm for three important noise fields: coherent (point source), diffused (spherically isotropic), and incoherent (noise signals generated at the sensors; e.g., amplifier noise, are assumed to be uncorrelated).

Coherent Noise Field

Assume a single stationary point source noise signal with PSD $\Phi_{n_sn_s}(k,\ell)$ and assume that $b_m(t)$ are slowly time-varying ATFs relating the noise source and the m-th sensor. Define,

$$
\boldsymbol{N}_s(k,\ell)=\boldsymbol{B}(k)N_s(k,\ell)\,,
$$

where

$$
\boldsymbol{B}^{T}(k)=(B_1(k)\;\;B_2(k)\;\;\cdots\;\;B_M(k))\,.
$$

The PSD matrix of the noise component at the sensors' signals is given by,

$$
\Phi_{N_sN_s}(k,\ell)=\Phi_{n_sn_s}(k,\ell)\boldsymbol{B}(k)\boldsymbol{B}^{H}(k)+\varepsilon\mathbf{I}\,,
$$

where $\mathbf{I}$ is an $M\times M$ identity matrix, and $\varepsilon\to 0$. The last term is added for stability reasons (see Appendix 47.A). For $\boldsymbol{B}(k)\neq\boldsymbol{A}(k)$, the achievable NR is infinite, i. e.,

$$
\Phi_{yy}^{n}(k,\ell)=0\quad\text{for}\quad\boldsymbol{B}(k)\neq\boldsymbol{A}(k)\,.
$$

Thus, perfect noise cancellation is achieved. The derivation of this result is given in Appendix 47.A. Note that this is not a surprising result, since for $M\geq 2$ the Wiener filter can entirely eliminate the noise component. This result is valid for all ATFs $\boldsymbol{B}(k)$ provided that $\boldsymbol{B}(k)\neq\boldsymbol{A}(k)$, i. e., the noise and the signal do not originate from the same point. If $\boldsymbol{B}(k)=\boldsymbol{A}(k)$ the noise and desired signal are indistinguishable and no NR is expected. If $\boldsymbol{B}(k)\neq\boldsymbol{A}(k)$ the proposed algorithm can eliminate any point source noise signal as good as the D-GSC can eliminate a directional noise signal in the delay-only propagation case (see [47.40]).

It is also interesting to explore the contribution of the FBF block to the NR,

$$
\begin{aligned}
&\Phi_{FBF}^{n}(k,\ell)\\
&=\frac{\tilde{A}^{H}(k)\Phi_{N_sN_s}(k,\ell)\tilde{A}(k)}{\|\tilde{A}(k)\|^4}\\
&=\frac{\tilde{A}^{H}(k)\Phi_{n_sn_s}(k,\ell)\boldsymbol{B}(k)\boldsymbol{B}^{H}(k)\tilde{A}(k)}{\|\tilde{A}(k)\|^4}\\
&=\frac{\Phi_{n_sn_s}(k,\ell)}{\|\tilde{A}(k)\|^4}\tilde{A}^{H}(k)\boldsymbol{B}(k)\left[\tilde{A}^{H}(k)\boldsymbol{B}(k)\right]^{H}\,.
\end{aligned}
$$

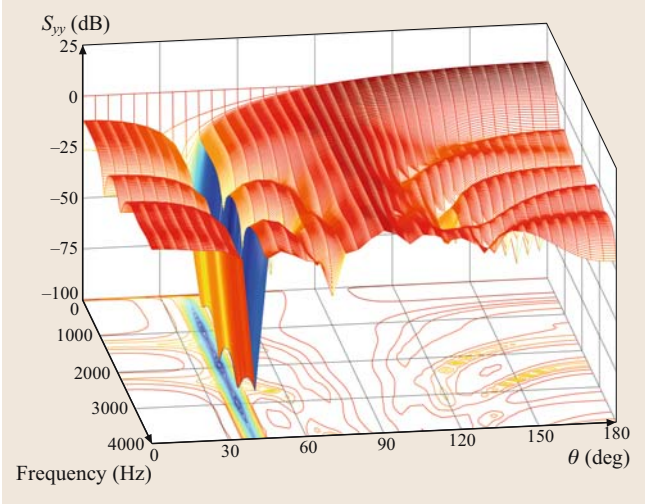


Fig. 47.7 Array output for directional noise field for linear array with $M=5$ sensors for delay-only ATFs

Although the FBF combines the desired signal coherently, it does not necessarily improves the SNR. The infinite NR is due to the ANC branch.

The expected NR in the simple case of delay-only ATFs is shown in Fig. 47.7. We optimize a linear array of $M = 5$ microphones to cancel noise impinging on the array from the direction $\theta = 40°$. We present the output signal of the array as a function of the frequency and the array steering angle θ. It is clearly shown that the main lobe is maintained (i. e., low distortion), while a null is constructed at all frequencies at the noise angle. The main lobe is wider in the lower-frequency band. This result is in good agreement with the theory, since at $\omega = 0\,\mathrm{rad/s}$ there is no phase difference between the signals at the sensors. Similar results were obtained by *Bitzer* et al. [47.40,41]. The general ATFs case is further explored in [47.77,97].

Diffused Noise Field

In highly reverberant acoustical environment, such as a car enclosure, the noise field tends to be diffused (see for instance [47.42,98]). A diffused noise source is assumed to be equidistributed on a sphere in the far field of the array. The cross-coherence function between signals received by two sensors (i,j) with distance d_{ij} is given by

$$\Gamma_{N_i N_j}(k) = \frac{\Phi_{N_i N_j}(k)}{\sqrt{\Phi_{N_i N_i}(k)\Phi_{N_j N_j}(k)}} = \frac{\sin(kd_{ij}/c)}{kd_{ij}/c} ,$$

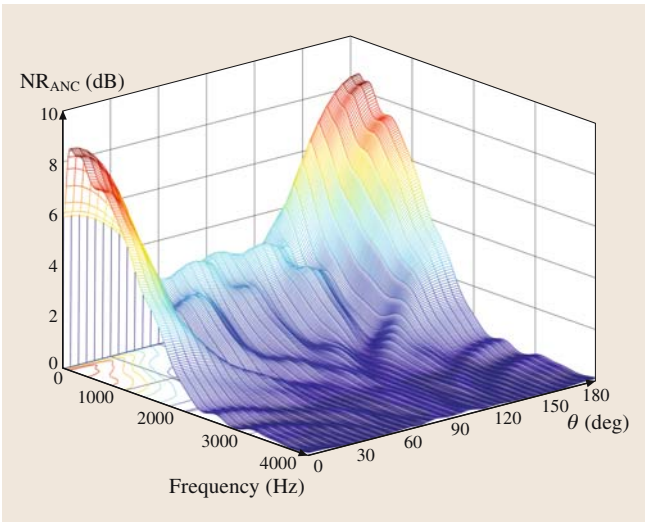


Fig. 47.8 Extra noise reduction of noise cancelling branch for diffused noise field using a linear array with $M = 5$ sensors and delay-only ATFs

where c is the speed of sound [47.98]. Thus, the coherence matrix is given by,

$$\Gamma(k) = \begin{pmatrix} 1 & \Gamma_{N_1 N_2}(k) & \cdots & \Gamma_{N_1 N_M}(k) \\ \Gamma_{N_2 N_1}(k) & 1 & \cdots & \Gamma_{N_2 N_M}(k) \\ \vdots & & \ddots & \vdots \\ \Gamma_{N_M N_1}(k) & & & 1 \end{pmatrix} .$$

The noise PSD at the sensors input is

$$\mathbf{\Phi}_{\boldsymbol{N}_\mathrm{s}\boldsymbol{N}_\mathrm{s}}(k,\ell) = \mathbf{\Phi}_{n_\mathrm{s}n_\mathrm{s}}(k,\ell)\Gamma(k) .$$

Using (47.120), the noise PSD at the FBF output is given by

$$\mathbf{\Phi}^{\mathrm{n}}_{\mathrm{FBF}}(k,\ell) = \frac{\phi_{\mathrm{nn}}(k,\ell)\tilde{\boldsymbol{A}}^{\mathrm{H}}(k)\Gamma(k)\tilde{\boldsymbol{A}}(k)}{\|\tilde{\boldsymbol{A}}(k)\|^4} .$$

The extra NR obtained by the ANC is given by

$$\begin{aligned} &\mathrm{NR}_{\mathrm{ANC}}(k,\ell) \\ &= \left\{1 - \left[\tilde{\boldsymbol{A}}^{\mathrm{H}}(k)\Gamma(k)\mathcal{H}(k)\left[\mathcal{H}^{\mathrm{H}}(k)\Gamma(k)\mathcal{H}(k)\right]^{-1}\right.\right. \\ &\quad \left.\left. \times \mathcal{H}^{\mathrm{H}}(k)\Gamma(k)\tilde{\boldsymbol{A}}(k)[\tilde{\boldsymbol{A}}^{\mathrm{H}}(k)\Gamma(k)\tilde{\boldsymbol{A}}(k)]^{-1}\right]\right\}^{-1} . \end{aligned} \tag{47.122}$$

This expression depends on the RTFs $\tilde{\boldsymbol{A}}(k)$, assumed to be error free, and on the coherence function $\Gamma(k)$.

The same $M = 5$ microphone array used for the directional noise case is now used for the diffused noise field. In Fig. 47.8 we show the extra NR obtained by the ANC for various steering angles and for the entire frequency band. It is clear that almost no NR is obtained in the high-frequency band and only relatively low NR in the low-frequency band. The obtained results are in accordance with the results in [47.39,42].

Incoherent Noise Field

For incoherent noise field we assume that the noise at the sensors has no spatial correlation.

$$\mathbf{\Phi}_{\boldsymbol{N}_\mathrm{s}\boldsymbol{N}_\mathrm{s}}(k,\ell) = \mathbf{\Phi}_{n_\mathrm{s}n_\mathrm{s}}(k,\ell)\mathbf{I} ,$$

where $\mathbf{I}$ is an $M \times M$ identity matrix. Using (47.119) with perfect knowledge of the RTFs, i. e., $\hat{\tilde{\boldsymbol{A}}}(k) = \tilde{\boldsymbol{A}}(k)$, and with the prespecified $\mathbf{\Phi}_{\boldsymbol{N}_\mathrm{s}\boldsymbol{N}_\mathrm{s}}(k,\ell)$ we obtain,

$$\begin{aligned} &\mathbf{\Phi}^{\mathrm{n}}_{\mathrm{yy}}(k,\ell) \\ &= \frac{\mathbf{\Phi}_{n_\mathrm{s}n_\mathrm{s}}(k,\ell)}{\|\tilde{\boldsymbol{A}}(k)\|^4}\tilde{\boldsymbol{A}}^{\mathrm{H}}(k) \\ &\quad \times \left\{\mathbf{I} - \mathcal{H}(k)\left[\mathcal{H}^{\mathrm{H}}(k)\mathcal{H}(k)\right]^{-1}\mathcal{H}^{\mathrm{H}}(k)\right\}\tilde{\boldsymbol{A}}(k) . \end{aligned}$$

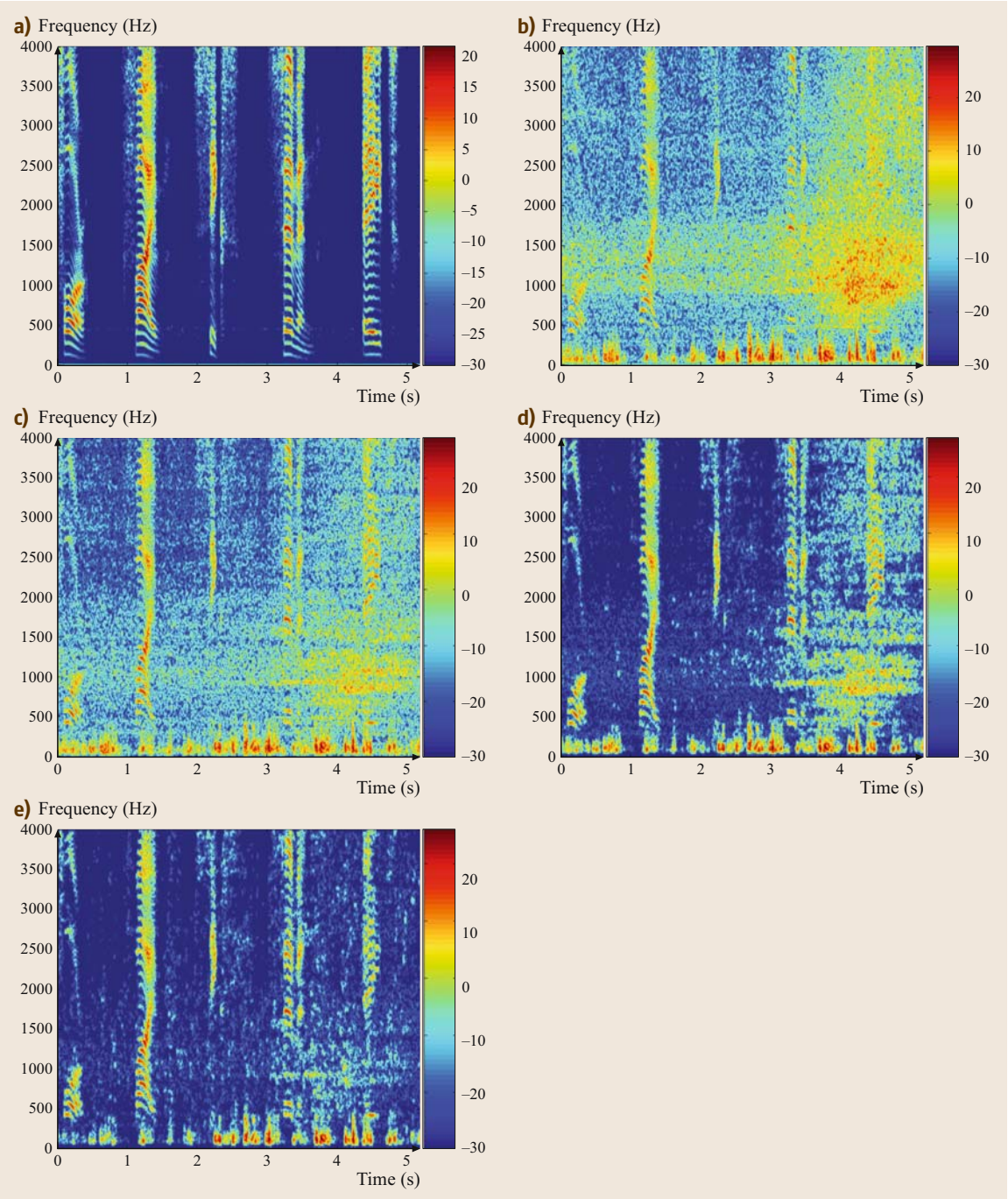


Fig. 47.9 (a) Original clean speech signal at microphone #1: "Five six seven eight nine". (b) Noisy signal at microphone #1. (c) TF-GSC output. (d) Single-channel postfiltering output. (e) Multichannel postfiltering.

It can be easily verified that,

$$\tilde{\boldsymbol{A}}^{\mathrm{H}}(k)\mathcal{H}(k) = \mathbf{0}_{1\times(M-1)} \,.$$

Furthermore, as $\mathcal{H}^{\mathrm{H}}(k)\mathcal{H}(k)$ is a positive matrix, its inverse always exists. Thus, the contribution of the noise cancelling branch is zero, and the NR is only attributed to the FBF. The noise power at the output is thus,

$$\begin{aligned}\boldsymbol{\Phi}^{\mathrm{n}}_{\mathrm{yy}}(k,\ell) = \boldsymbol{\Phi}^{\mathrm{n}}_{\mathrm{FBF}}(k,\ell) &= \boldsymbol{\Phi}_{n_{\mathrm{s}}n_{\mathrm{s}}}(k,\ell)\frac{\tilde{\boldsymbol{A}}^{\mathrm{H}}(k)\tilde{\boldsymbol{A}}(k)}{\|\tilde{\boldsymbol{A}}(k)\|^4} \\ &= \frac{\boldsymbol{\Phi}_{n_{\mathrm{s}}n_{\mathrm{s}}}(k,\ell)}{\|\tilde{\boldsymbol{A}}(k)\|^2} \,.\end{aligned}$$

Again, no NR is guaranteed by this structure, and the result depends on the RTFs involved.

In the case of delay-only ATFs, the FBF branch becomes a simple delay-and-sum beamformer, thus the expected NR is M, the number of sensors.

47.8 Experimental Results

In this section, we compare the performance of a system, consisting of the TF-GSC and multichannel postfilter, to a system consisting of a TF-GSC and a single-channel postfilter.

A linear array, consisting of four microphones with 5 cm spacing, is mounted in a car on the visor. Clean speech signals are recorded at a sampling rate of 8 kHz in the absence of background noise (standing car, silent environment). A car noise signal is recorded while the car speed is about 60 km/h, and the window next to the driver is slightly open (about 5 cm; the other windows are closed). The input microphone signals are generated by mixing the speech and noise signals at various SNR levels in the range $[-5, 10]$ dB.

Offline TF-GSC beamforming [47.38] is applied to the noisy multichannel signals, and its output is enhanced using the OM-LSA estimator [47.93]. The result is referred to as single-channel postfiltering output. Alternatively, the proposed TF-GSC and multichannel postfiltering is applied to the noisy signals. A subjective comparison between multichannel and single-channel postfiltering was conducted using speech sonograms and validated by informal listening tests. Typical examples of speech sonograms are presented in Fig. 47.9. For audio samples please refer to [47.99]. The noise PSD at the beamformer output varies substantially due to the residual interfering components of speech, wind blows, and passing cars. The TF-GSC output is characterized by a high level of noise. Single-channel postfiltering suppresses pseudostationary noise components, but is inefficient at attenuating the transient noise components. By contrast, the system which consists a TF-GSC and multichannel postfilter achieves superior noise attenuation, while preserving the desired source components. This is verified by subjective informal listening tests.

47.9 Summary

In this chapter, we concentrated on the GSC beamformer, and presented a comprehensive study of its components. We showed, that the GSC structure is closely related to other array optimization criteria, such as the Wiener filter. We described multimicrophone postfilters, based on either the MMSE or the log-spectral estimation criteria, which are designed for improving the amount of obtainable NR, with minimal degradation of speech quality. The robustness of the GSC structure to imperfect estimation of its components was analyzed. Various methods were proposed for increasing the robustness of the GSC, and especially, for avoiding leakage of the desired signal into the noise reference signals. Finally, the performance of the TF-GSC was theoretically analyzed and experimentally evaluated under nonstationary noise conditions.

47.A Appendix: Derivation of the Expected Noise Reduction for a Coherent Noise Field

For clarity of the exposition we will omit the time and frequency dependence in the derivation. Recall (47.119) and define

$$\mathcal{X}(k,\ell) \triangleq \mathcal{X} = \boldsymbol{\Phi}_{\boldsymbol{N}_\mathrm{s}\boldsymbol{N}_\mathrm{s}}\mathcal{H}\big(\mathcal{H}^\mathrm{H}\boldsymbol{\Phi}_{\boldsymbol{N}_\mathrm{s}\boldsymbol{N}_\mathrm{s}}\mathcal{H}\big)^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{\Phi}_{\boldsymbol{N}_\mathrm{s}\boldsymbol{N}_\mathrm{s}}\,. \tag{47.A1}$$

Denote,

$$\mathcal{X} \triangleq \mathcal{K}\times\mathcal{L}\times\mathcal{M}\,,$$

where, $\mathcal{K} = \boldsymbol{\Phi}_{\boldsymbol{N}_\mathrm{s}\boldsymbol{N}_\mathrm{s}}\mathcal{H}$, $\mathcal{L} = (\mathcal{H}^\mathrm{H}\boldsymbol{\Phi}_{\boldsymbol{N}_\mathrm{s}\boldsymbol{N}_\mathrm{s}}\mathcal{H})^{-1}$, and $\mathcal{M} = \mathcal{H}^\mathrm{H}\boldsymbol{\Phi}_{\boldsymbol{N}_\mathrm{s}\boldsymbol{N}_\mathrm{s}}$. Thus, $\mathcal{X}$ is a multiplication of three terms.

Starting from $\mathcal{L}$ and using the detailed noise structure,

$$\begin{aligned}\mathcal{L} &= \big[\mathcal{H}^\mathrm{H}\big(\phi_\mathrm{nn}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}+\varepsilon\mathbf{I}\big)\mathcal{H}\big]^{-1}\\ &= \big[\phi_\mathrm{nn}(\mathcal{H}^\mathrm{H}\boldsymbol{B})(\mathcal{H}^\mathrm{H}\boldsymbol{B})^\mathrm{H}+\varepsilon\mathcal{H}^\mathrm{H}\mathcal{H}\big]^{-1}\,.\end{aligned}$$

If $\boldsymbol{B}=\boldsymbol{A}$, i. e., the noise source is located exactly at the desired signal position, then $\mathcal{H}^\mathrm{H}\boldsymbol{B}=0$, and the calculation of the inverse is straightforward, yielding $\mathcal{L}=(\varepsilon\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}$, $\mathcal{K}=\varepsilon\mathcal{H}$, and $\mathcal{M}=\varepsilon\mathcal{H}^\mathrm{H}$. Collecting all terms we obtain, $\mathcal{X}=\frac{\varepsilon^2}{\varepsilon}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\overset{\varepsilon\to0}{\longrightarrow}0$, i. e., the signal at the BM output is zero, as expected. The total noise part of the output is given by,

$$\begin{aligned}\boldsymbol{\Phi}^\mathrm{n}_\mathrm{yy} &= \boldsymbol{\Phi}^\mathrm{n}_\mathrm{FBF}\\ &= \frac{\hat{\boldsymbol{A}}^\mathrm{H}\boldsymbol{\Phi}_{\boldsymbol{N}_\mathrm{s}\boldsymbol{N}_\mathrm{s}}\hat{\boldsymbol{A}}}{\|\hat{\boldsymbol{A}}\|^4} = \frac{\hat{\boldsymbol{A}}^\mathrm{H}\big(\phi_\mathrm{nn}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}+\varepsilon\mathbf{I}\big)\hat{\boldsymbol{A}}}{\|\hat{\boldsymbol{A}}\|^4}\\ &\overset{\varepsilon\to0}{\longrightarrow}\phi_{nn}|A_1|^2\,,\end{aligned}$$

where the last transition is due to $\boldsymbol{B}=\boldsymbol{A}$. This is exactly the no-distortion result obtained in Sect. 47.7.2, for the desired signal direction.

Table 47.4 Twelve terms used for calculating $\mathcal{X}=\mathcal{K}\mathcal{L}\mathcal{M}$

$\mathcal{K}_1\mathcal{L}_1\mathcal{M}_1$	$= \frac{1}{\varepsilon}\boldsymbol{\Phi}^2_\mathrm{nn}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}$;	(I)
$\mathcal{K}_1\mathcal{L}_2\mathcal{M}_1$	$= -\frac{1}{\varepsilon}\boldsymbol{\Phi}^3_\mathrm{nn}\times\frac{\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}}{\phi_\mathrm{nn}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}}$	(II)
	$= -\frac{1}{\varepsilon}\boldsymbol{\Phi}^2_\mathrm{nn}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}$;	
$\mathcal{K}_1\mathcal{L}_3\mathcal{M}_1$	$= \frac{\boldsymbol{\Phi}^3_\mathrm{nn}\boldsymbol{B}(\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B})^2\boldsymbol{B}^\mathrm{H}}{(\phi_\mathrm{nn}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B})^2}$	(III)
	$= \phi_\mathrm{nn}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}$;	
$\mathcal{K}_2\mathcal{L}_1\mathcal{M}_1$	$= \phi_\mathrm{nn}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}$;	(IV)
$\mathcal{K}_2\mathcal{L}_2\mathcal{M}_1$	$= -\boldsymbol{\Phi}^2_\mathrm{nn}\frac{\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}}{\phi_\mathrm{nn}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}}$	(V)
	$= -\phi_\mathrm{nn}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}$;	
$\mathcal{K}_2\mathcal{L}_3\mathcal{M}_1$	$= \varepsilon\frac{\boldsymbol{\Phi}^2_\mathrm{nn}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}}{(\phi_\mathrm{nn}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B})^2}$	(VI)
	$= \varepsilon\frac{\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}}{\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}}$;	
$\mathcal{K}_1\mathcal{L}_1\mathcal{M}_2$	$= \phi_\mathrm{nn}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}$;	(VII)
$\mathcal{K}_1\mathcal{L}_2\mathcal{M}_2$	$= -\boldsymbol{\Phi}^2_\mathrm{nn}\frac{\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}}{\phi_\mathrm{nn}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}}$	(VIII)
	$= -\phi_\mathrm{nn}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}$;	
$\mathcal{K}_1\mathcal{L}_3\mathcal{M}_2$	$= \varepsilon\frac{\boldsymbol{\Phi}^2_\mathrm{nn}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}}{(\phi_\mathrm{nn}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B})^2}$	(IX)
	$= \varepsilon\frac{\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}}{\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}}$;	
$\mathcal{K}_2\mathcal{L}_1\mathcal{M}_2$	$= \varepsilon\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}$;	(X)
$\mathcal{K}_2\mathcal{L}_2\mathcal{M}_2$	$= -\varepsilon\frac{\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}}{\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}}$;	(XI)
$\mathcal{K}_2\mathcal{L}_3\mathcal{M}_2$	$= \varepsilon^2\frac{\phi_\mathrm{nn}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}}{(\phi_\mathrm{nn}\boldsymbol{B}^\mathrm{H}\mathcal{H}(\mathcal{H}^\mathrm{H}\mathcal{H})^{-1}\mathcal{H}^\mathrm{H}\boldsymbol{B})^2}$.	(XII)

For the general case, $\boldsymbol{B} \neq \boldsymbol{A}$, we use the *matrix inversion lemma*, yielding:

$$\mathcal{L} = \frac{1}{\varepsilon}(\mathcal{H}^{\mathrm{H}}\mathcal{H})^{-1} - \frac{\frac{1}{\varepsilon^2}\phi_{\mathrm{nn}}(\mathcal{H}^{\mathrm{H}}\mathcal{H})^{-1}\mathcal{H}^{\mathrm{H}}\boldsymbol{B}\boldsymbol{B}^{\mathrm{H}}\mathcal{H}(\mathcal{H}^{\mathrm{H}}\mathcal{H})^{-1}}{1+\frac{1}{\varepsilon}\phi_{\mathrm{nn}}\boldsymbol{B}^{\mathrm{H}}\mathcal{H}(\mathcal{H}^{\mathrm{H}}\mathcal{H})^{-1}\mathcal{H}^{\mathrm{H}}\boldsymbol{B}} .$$

Now, using the approximation $\frac{1}{1+\mu} \approx 1-\mu$, for $\mu \to 0$ (μ properly defined), yields,

$$\mathcal{L} = \frac{1}{\varepsilon}(\mathcal{H}^{\mathrm{H}}\mathcal{H})^{-1} - \frac{\frac{1}{\varepsilon}\phi_{\mathrm{nn}}(\mathcal{H}^{\mathrm{H}}\mathcal{H})^{-1}\mathcal{H}^{\mathrm{H}}\boldsymbol{B}\boldsymbol{B}^{\mathrm{H}}\mathcal{H}(\mathcal{H}^{\mathrm{H}}\mathcal{H})^{-1}}{\phi_{\mathrm{nn}}\boldsymbol{B}^{\mathrm{H}}\mathcal{H}(\mathcal{H}^{\mathrm{H}}\mathcal{H})^{-1}\mathcal{H}^{\mathrm{H}}\boldsymbol{B}} + \frac{\phi_{\mathrm{nn}}(\mathcal{H}^{\mathrm{H}}\mathcal{H})^{-1}\mathcal{H}^{\mathrm{H}}\boldsymbol{B}\boldsymbol{B}^{\mathrm{H}}\mathcal{H}(\mathcal{H}^{\mathrm{H}}\mathcal{H})^{-1}}{\left(\phi_{\mathrm{nn}}\boldsymbol{B}^{\mathrm{H}}\mathcal{H}(\mathcal{H}^{\mathrm{H}}\mathcal{H})^{-1}\mathcal{H}^{\mathrm{H}}\boldsymbol{B}\right)^2} . \quad (47.A2)$$

Now, calculating $\mathcal{X}$,

$$\begin{aligned}\mathcal{X} &= \mathcal{K}\mathcal{L}\mathcal{M} = \boldsymbol{\Phi}_{NN}\mathcal{H}\mathcal{L}\mathcal{H}^{\mathrm{H}}\boldsymbol{\Phi}_{NN} \\ &= \left(\phi_{\mathrm{nn}}\boldsymbol{B}\boldsymbol{B}^{\mathrm{H}}+\varepsilon\mathbf{I}\right)\mathcal{H}\mathcal{L}\mathcal{H}^{\mathrm{H}}\left(\phi_{\mathrm{nn}}\boldsymbol{B}\boldsymbol{B}^{\mathrm{H}}+\varepsilon\mathbf{I}\right) \\ &\triangleq (\mathcal{K}_1+\mathcal{K}_2)\mathcal{H}(\mathcal{L}_1+\mathcal{L}_2+\mathcal{L}_3)\mathcal{H}^{\mathrm{H}}(\mathcal{M}_1+\mathcal{M}_2) , \end{aligned} \quad (47.A3)$$

with the obvious definitions of $\mathcal{K}_1$, $\mathcal{K}_2$, $\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_3$, $\mathcal{M}_1$, and $\mathcal{M}_2$. Opening the brackets we have twelve terms given in Table 47.4.

Note, that terms I and II, terms IV and V, and terms VII and VIII eliminate each other, and that terms VI, IX, X, XI, and XII vanish as ε approaches to zero. Only term III is left, i. e., $\mathcal{X} = \phi_{\mathrm{nn}}BB^{\mathrm{H}}$. Substituting $\mathcal{X}$ into (47.119) we have

$$\boldsymbol{\Phi}_{\mathrm{yy}}^{\mathrm{n}} = \frac{\tilde{\boldsymbol{A}}^{\mathrm{H}}\boldsymbol{\Phi}_{N_{\mathrm{s}}N_{\mathrm{s}}}\tilde{\boldsymbol{A}}}{\|\tilde{\boldsymbol{A}}\|^4} - \frac{\tilde{\boldsymbol{A}}^{\mathrm{H}}\left(\phi_{\mathrm{nn}}BB^{\mathrm{H}}\right)\tilde{\boldsymbol{A}}}{\|\tilde{\boldsymbol{A}}\|^4} = 0 . \quad (47.A4)$$

47.B Appendix: Equivalence Between Maximum SNR and LCMV Beamformers

The LCMV beamformer is given by

$$W^{\mathrm{LCMV}} = \frac{\left(\phi_{\mathrm{ss}}\boldsymbol{A}\boldsymbol{A}^{\mathrm{H}}+\boldsymbol{\Phi}_{NN}\right)^{-1}\boldsymbol{A}}{\boldsymbol{A}^{\mathrm{H}}\left(\phi_{\mathrm{ss}}\boldsymbol{A}\boldsymbol{A}^{\mathrm{H}}+\boldsymbol{\Phi}_{NN}\right)^{-1}\boldsymbol{A}} . \quad (47.B1)$$

Using the matrix inversion lemma we have in the denominator

$$\begin{aligned}&\boldsymbol{A}^{\mathrm{H}}\left(\boldsymbol{\Phi}_{NN}+\phi_{\mathrm{ss}}\boldsymbol{A}\boldsymbol{A}^{\mathrm{H}}\right)^{-1}\boldsymbol{A} \\ &= \boldsymbol{A}^{\mathrm{H}}\left(\boldsymbol{\Phi}_{NN}^{-1}-\frac{\phi_{\mathrm{ss}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}}{1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}}\right)\boldsymbol{A} \\ &= \boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}-\boldsymbol{A}^{\mathrm{H}}\frac{\phi_{\mathrm{ss}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}}{1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}}\boldsymbol{A} \\ &= \boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}-\phi_{\mathrm{ss}}\frac{\left(\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}\right)^2}{1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}} \\ &= \frac{\left(1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}\right)\left(\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}\right)}{1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}} \\ &\quad -\frac{\phi_{\mathrm{ss}}\left(\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}\right)^2}{1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}} \\ &= \frac{\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}}{1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}} \end{aligned} \quad (47.B2)$$

Similarly, we have in the numerator

$$\begin{aligned}&\left(\phi_{\mathrm{ss}}\boldsymbol{A}\boldsymbol{A}^{\mathrm{H}}+\boldsymbol{\Phi}_{NN}\right)^{-1}\boldsymbol{A} \\ &= \boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}-\frac{\phi_{\mathrm{ss}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}}{1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}} \\ &= \frac{\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}\left(1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}\right)}{1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}} \\ &\quad -\frac{\phi_{\mathrm{ss}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}}{1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}} \\ &= \frac{\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}}{1+\phi_{\mathrm{ss}}\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}} . \end{aligned} \quad (47.B3)$$

Dividing (47.B3) by (47.B2) we have

$$W^{\mathrm{LCMV}} = \frac{\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}}{\boldsymbol{A}^{\mathrm{H}}\boldsymbol{\Phi}_{NN}^{-1}\boldsymbol{A}} \equiv W^{\mathrm{MSNR}} . \quad (47.B4)$$

References

47.1 S. Haykin: *Adaptive Filter Theory*, 4th edn. (Prentice Hall, Upper Saddle River 2002)
47.2 B. Widrow, S.D. Stearns: *Adaptive Signal Processing* (Prentice-Hall, Upper Saddle River 1985)
47.3 D. Monzingo, T. Miller: *Introduction to Adaptive Arrays* (Wiley, New York 1980)
47.4 H.L.V. Trees: *Detection, Estimation, and Modulation Theory*, Vol. IV (Wiley, New York 2002)
47.5 B.V. Veen, K. Buckley: Beamforming: A versatile approach to spatial filtering, IEEE Acoust. Speech Signal Process. Mag. **5**(2), 4–24 (1988)
47.6 H. Krim, M. Viberg: Two decades of array signal processing research: The parametric approach, IEEE Signal Proc. Mag. **13**, 67–94 (1996)
47.7 H. Cox, R. Zeskind, M. Owen: Robust Adaptive Beamforming, IEEE Trans. Acoust. Speech Signal Process. **35**(10), 1365–1376 (1987)
47.8 S.L. Gay, J. Benesty: *Acoustic Signal Processing for Telecommunication* (Kluwer Academic, Dordrecht 2001)
47.9 M.S. Brandstein, D.B. Ward: *Microphone Arrays: Signal Processing Techniques and Applications* (Springer, Berlin, Heidelberg 2001)
47.10 J. Benesty, Y. Huang (Eds.): *Adaptive Signal Processing: Applications to Real-World Problems* (Springer, Berlin, Heidelberg 2003)
47.11 E. Hänsler, G. Schmidt: *Acoustic Echo and Noise Control: A Pracical Approach* (Wiley, New York 2004)
47.12 J. Benesty, S. Makino, J. Chen: *Speech Enhancement* (Springer, Berlin, Heidelberg 2005)
47.13 W. Herbordt: *Sound Capture for Human/Machine Interfaces – Practical Aspects of Microphone Array Signal Processing* (Springer, Berlin, Heidelberg 2005)
47.14 S. Gannot, J. Benesty, J. Bitzer, I. Cohen, S. Doclo, R. Martin, S. Nordholm: Advances in multimicrophone speech processing, EURASIP J. Appl. Signal Process. **2006**(ID46357) (2006)
47.15 E. Jan, J. Flanagan: Microphone arrays for speech processing, International Symposium on Signals, Systems, and Electronics (ISSSE) (URSI, San Francisco 1995) pp. 373–376
47.16 E. Jan, J. Flanagan: Sound capture from spatial volumes: matched-filter processing of microphone arrays having randomly-distributed sensors, Proc. ICASSP (Atlanta, 1996) pp. 917–920
47.17 D. Rabinkin, R. Renomeron, J. Flanagan, D. Macomber: Optimal truncation time for matched filter array processing, Proc. ICASSP (Seattle, 1998) pp. 3269–3272
47.18 M. Sondhi, G. Elko: Adaptive optimization of microphone arrays under a nonlinear constraint, Proc. ICASSP, Vol. 11 (Tokyo, 1986) pp. 981–984
47.19 Y. Kaneda, J. Ohga: Adaptive microphone-array system for noise reduction, IEEE Trans. Acoust. Speech Signal Process. **34**(6), 1391–1400 (1986)
47.20 D.V. Compernolle: Switching adaptive filters for enhancing noisy and reverberant speech from microphone array recordings, Proc. ICASSP (Albuquerque, 1990) pp. 833–836
47.21 W. Kellermann: Acoustic echo cancellation for beamforming microphone arrays. In: *Microphone Arrays: Signal Processing Techniques and Applications*, ed. by M.S. Brandstein, D.B. Ward (Springer, Berlin, Heidelberg 2001) pp. 281–306
47.22 I. Claesson, S. Nordholm: A spatial filtering approach to robust adaptive beamforming, IEEE Trans. Antennas Propag. **40**(1), 1093–1096 (1992)
47.23 S. Nordholm, I. Claesson, N. Grbić: Optimal and adaptive microphone arrays for speech input in automobiles. In: *Microphone Arrays: Signal Processing Techniques and Applications*, ed. by M.S. Brandstein, D.B. Ward (Springer, Berlin, Heidelberg 2001) pp. 307–330
47.24 R. Martin: Small microphone arrays with postfilters for noise and acoustic echo reduction. In: *Microphone Arrays: Signal Processing Techniques and Applications*, ed. by M.S. Brandstein, D.B. Ward (Springer, Berlin, Heidelberg 2001) pp. 255–280
47.25 B. Widrow, J.R. Glover Jr., J.M. McCool, J. Kaunitz, C.S. Williams, R.H. Hearn, J.R. Zeidler, E. Dong Jr, R.C. Goodlin: Adaptive noise cancelling: Principals and applications, Proc. IEEE **63**(12), 1692–1716 (1975)
47.26 S. Doclo, M. Moonen: GSVD-based optimal filtering for single and multimicrophone speech enhancement, IEEE Trans. Speech Audio Process. **50**(9), 2230–2244 (2002)
47.27 S. Doclo, M. Moonen: Design of far-field and near-field broadband beamformers using eigenfilters, Signal Process. **83**(12), 2641–2673 (2003)
47.28 S. Doclo, M. Moonen: Multi-microphone noise reduction using recursive GSVD-based optimal filtering with ANC postprocessing stage, IEEE Trans. Speech Audio Process. **13**(1), 53–69 (2005)
47.29 A. Spriet, M. Moonen, J. Wouters: A multichannel subband GSVD approach for speech enhancement, Eur. Trans. Telecommun. **13**(2), 149–158 (2002), Special Issue on Acoustic Echo and Noise Control
47.30 G. Rombouts, M. Moonen: QRD-based unconstrained optimal filtering for acoustic noise reduction, Signal Process. **83**(9), 1889–1904 (2003)
47.31 G. Rombouts, M. Moonen: Fast QRD-lattice-based unconstrained optimal filtering for acoustic noise reduction, IEEE Trans. Speech Audio Process. **13**(6), 1130–1143 (2005)

47.32 J. Chen, J. Benesty, Y. Huang, S. Doclo: New insights into the noise reduction Wiener filter, IEEE Trans. Speech Audio Process. **14**(4), 1218–1234 (2006)

47.33 A. Spriet, M. Moonen, J. Wouters: Spatially pre-processed speech distortion weighted multi-channel Wiener filtering for noise reduction, Signal Process. **84**(12), 2367–2387 (2004)

47.34 S. Doclo, A. Spriet, J. Wouters, M. Moonen: Speech distortion weighted multichannel wiener filtering techniques for noise reduction. In: *Speech Enhancement*, Signals and Communication Technology, ed. by J. Benesty, S. Makino, J. Chen (Springer, Berlin, Heidelberg 2005) pp. 199–228

47.35 O.L. Frost: An algorithm for linearly constrained adaptive array processing, Proc. IEEE **60**(8), 926–935 (1972)

47.36 L.J. Griffiths, C.W. Jim: An alternative approach to linearly constrained adaptive beamforming, IEEE Trans. Antennas Propag. **30**(1), 27–34 (1982)

47.37 S. Affes, Y. Grenier: A source subspace tracking array of microphones for double talk situations, Proc. ICASSP (Munich, 1997) pp. 269–272

47.38 S. Gannot, D. Burshtein, E. Weinstein: Signal enhancement using beamforming and nonstationarity with applications to speech, IEEE Trans. Signal Proces. **49**(8), 1614–1626 (2001)

47.39 S. Nordholm, Y.H. Leung: Performance limits of the broadband generalized sidelobe cancelling structure in an isotropic noise field, J. Acoust. Soc. Am. **107**(2), 1057–1060 (2000)

47.40 J. Bitzer, K.U. Simmer, K.D. Kammeyer: Theoretical noise reduction limits of the generalized sidelobe canceller (GSC) for speech enhancement, Proc. ICASSP (Phoenix, 1999) pp. 2965–2968

47.41 J. Bitzer, K. Simmer, K. Kammeyer: Multichannel noise reduction – algorithms and theoretical limits, Proc. EUSIPCO, Vol. I (Rhodes, 1998) pp. 105–108

47.42 J. Bitzer, K.-D. Kammeyer, K. Simmer: An alternative implementation of the superdirective beamformer, Workshop on Application of Signal Processing to Audio and Acoustics (IEEE, New Paltz, NY 1999)

47.43 C. Marro, Y. Mahieux, K. Simmer: Analysis of noise reduction and dereverberation techniques based on microphone arrays with postfiltering, IEEE Trans. Speech Audio Process. **6**(3), 240–259 (1998)

47.44 S. Nordholm, I. Claesson, M. Dahl: Adaptive microphone array employing calibration signals: an analytical evaluation, IEEE Trans. Speech Audio Process. **7**(3), 241–252 (1999)

47.45 K.-C. Huarng, C.-C. Yeh: Performance analysis of derivative constraint adaptive arrays with pointing errors, IEEE Trans. Acoust. Speech Signal Process. **38**(2), 209–219 (1990)

47.46 S. Nordholm, I. Claesson, P. Eriksson: The broadband wiener solution for griffiths-jim beamformers, IEEE Trans. Signal Process. **40**(2), 474–478 (1992)

47.47 H. Cox: Resolving power and sensitivity to mismatch of optimum array processors, J. Acoust. Soc. Am. **54**(3), 771–785 (1973)

47.48 D. Brandwood: A complex gradient operator and its application in adaptive array theory, Proc. IEEE **130**(1 Parts F and H), 11–16 (1983)

47.49 G. Strang: *Linear Algebra and its Application*, 2nd edn. (Academic, New York 1980)

47.50 R. Crochiere: A weighted overlap-add method for short-time Fourier analysis/synthesis, IEEE Trans. Acoust. Speech Signal Process. **28**(1), 99–102 (1980)

47.51 J. Shynk: Frequency-domain and multirate and adaptive filtering, IEEE Signal Process. Mag. **9**(1), 14–37 (1992)

47.52 J. Allen, D. Berkley: Image method for efficiently simulating small-room acoustics, J. Acoust. Soc. Am. **65**(4), 943–950 (1979)

47.53 S. Doclo, M. Moonen: Combined frequency-domain dereverberation and noise reduction technique for multi-microphone speech enhancement, Int. Workshop Acoust. Echo Noise Control (IWAENC) (Darmstadt 2001) pp. 31–34

47.54 B. Yang: Projection approximation subspace tracking, IEEE Trans. Signal Process. **43**(1), 95–107 (1995)

47.55 O. Hoshuyama, A. Sugiyama: Robust adaptive beamforming. In: *Microphone Arrays*, ed. by M. Brandstein, D. Ward (Springer, Berlin, Heidelberg 2001) pp. 87–109

47.56 E. Weinstein, M. Feder, A. Oppenheim: Multichannel signal separation by decorrelation, IEEE Trans. Speech Audio Process. **1**(4), 405–413 (1993)

47.57 S. Gannot: *Array Processing of Nonstationary Signals with Application to Speech* (Tel-Aviv University, Tel-Aviv 2000), available on line, http://www.biu.ac.il/ gannot

47.58 C. Knapp, G. Carter: The generalized correlation method for estimation of time delay, IEEE Trans. Acoust. Speech Signal Process. **24**(4), 320–327 (1976)

47.59 T. Dvorkind, S. Gannot: Time difference of arrival estimation of speech source in a noisy and reverberant environment, Signal Process. **85**(1), 177–204 (2005)

47.60 J. Chen, J. Benesty, Y. Huang: Time delay estimation in room acoustic environments: an overview, EURASIP J. Appl. Signal Process. **26**(503), 19 (2006)

47.61 I. Cohen: Relative transfer function identification using speech signals, IEEE Trans. Speech Audio Process. **12**(5), 451–459 (2004)

47.62 I. Cohen: Identification of speech source coupling between sensors in reverberant noisy environments, IEEE Signal Process. Lett. **11**(7), 613–616 (2004)

47.63 O. Shalvi, E. Weinstein: System identification using nonstationary signals, IEEE Trans. Signal Process. **44**(8), 2055–2063 (1996)

47.64 D.G. Manolakis, V.K. Ingle, S.M. Kogan: *Statistical and Adaptive Signal Processing: Spectral Estima-*

tion, Signal Modeling, Adaptive Filtering and Array Processing (McGraw-Hill, Singapore 2000)
47.65 R. Martin: Noise power spectral density estimation based on optimal smoothing and minimum statistics, IEEE Trans. Speech Audio Process. **9**(5), 504–512 (2001)
47.66 I. Cohen, B. Berdugo: Noise estimation by minima controlled recursive averaging for robust speech enhancement, IEEE Signal Process. Lett. **9**(1), 12–15 (2002)
47.67 I. Cohen: Noise spectrum estimation in adverse environments: Improved minima controlled recursive averaging, IEEE Trans. Speech Audio Process. **11**(5), 466–475 (2003)
47.68 S. Nordholm, H.Q. Dam, N. Grbić, S.Y. Low: Adaptive microphone array employing spatial quadratic soft constraints and spectral shaping. In: *Speech Enhancement*, Signals and Communication Technology, ed. by J. Benesty, S. Makino, J. Chen (Springer, Berlin, Heidelberg 2005) pp. 229–246
47.69 O. Hoshuyama, A. Sugiyama, A. Hirano: A robust adaptive beamformer for microphone arrays with a blocking matrix using constrained adaptive filters, IEEE Trans. Signal Process. **47**(10), 2677–2684 (1999)
47.70 C. Fancourt, L. Parra: The generalized sidelobe decorrelator, Proc. IEEE Workshop on Applications of Signal Process. to Audio and Acoustics (New Paltz, NY 2001) pp. 167–170
47.71 W. Herbordt, W. Kellermann: Computationally efficient frequency-domain robust generalized sidelobe canceller, Int. Workshop Acoust. Echo Noise Control (IWAENC) (Darmstadt 2001) pp. 51–54
47.72 W. Herbordt, W. Kellermann: Adaptive beamforming for audio signal acquisition. In: *Adaptive Signal Processing: Applications to Real-World Problems*, Signals and Communication Technology, ed. by J. Benesty, Y. Huang (Springer, Berlin, Heidelberg 2003) pp. 155–194
47.73 S. Doclo, M. Moonen: On the output SNR of the speech-distortion weighted multichannel Wiener filter, IEEE Signal Process. Lett. **11**(12), 809–811 (2005)
47.74 A. Spriet, M. Moonen, J. Wouters: Stochastic gradient-based implementation of spatially preprocessed speech distortion weighted multichannel wiener filtering for noise reduction in hearing aids, IEEE Trans. Signal Process. **53**(3), 911–925 (2005)
47.75 A. Spriet, M. Moonen, J. Wouters: Robustness analysis of multichannel wiener filtering and generalized sidelobe cancellation for multimicrophone noise reduction in hearing aid applications, IEEE Trans. Speech Audio Process. **13**(4), 487–503 (2005)
47.76 S. Low, S. Nordholm, R. Togneri: Convolutive blind signal separation with post-processing, IEEE Trans. Speech Audio Process. **12**(5), 539–548 (2004)
47.77 S. Gannot, D. Burshtein, E. Weinstein: Analysis of the power spectral deviation of the general transfer function GSC, IEEE Trans. Signal Process. **52**(4), 1115–1121 (2004)
47.78 K.U. Simmer, J. Bitzer, C. Marro: Post-filtering techniques. In: *Microphone Arrays: Signal Processing Techniques and Applications*, ed. by M.S. Brandstein, D.B. Ward (Springer, Berlin, Heidelberg 2001) pp. 39–60
47.79 R. Zelinski: A Microphone array with adaptive post-filtering for noise reduction in reverberant rooms, IEEE ICASSP (New York, 1988) pp. 2578–2581
47.80 R. Zelinski: Noise reduction based on microphone array with LMS adaptive post-filtering, Electron. Lett. **26**(24), 2036–2581 (1990)
47.81 S. Fischer, K.U. Simmer: An adaptive microphone array for hands-free communication, Int. Workshop Acoust. Echo Noise Control (IWAENC) (Røros, 1995) pp. 44–47
47.82 S. Fischer, K.U. Simmer: Beamforming microphone arrays for speech acquisition in noisy environments, Speech Commun. **20**(3–4), 215–227 (1996)
47.83 S. Fischer, K.-D. Kammeyer: Broadband beamforming with adaptive postfiltering for speech acquisition in noisy environment, Proc. ICASSP, Vol. 1 (Munich, 1997) pp. 359–362
47.84 J. Meyer, K.U. Simmer: Multi-channel speech enhancement in a car environment using Wiener filtering and spectral subtraction, Proc. ICASSP (Munich, 1997) pp. 21–24
47.85 K.U. Simmer, S. Fischer, A. Wasiljeff: Suppression of coherent and incoherent noise using a microphone array, Annales des Télécommun. **49**(7–8), 439–446 (1994)
47.86 J. Bitzer, K. Simmer, K.-D. Kammeyer: Multimicrophone noise reduction by post-filter and superdirective beamformer, Int. Workshop Acoust. Echo Noise Control (IWAENC) (Pocono Manor, 1999) pp. 100–103
47.87 J. Bitzer, K.U. Simmer, K.-D. Kammeyer: Multimicrophone noise reduction techniques as front-end devices for speech recognition, Speech Commun. **34**, 3–12 (2001)
47.88 K.U. Simmer, A. Wasiljeff: Adptive microphone arrays for noise supression in the frequency domain, Proc. Second Cost 229 Workshop on Adaptive Algorithms in Communications (Bordeaux, 1992) pp. 185–194
47.89 I. McCowan, H. Bourlard: Microphone array post-filter for diffuse noise field, Proc. ICASSP, Vol. 1. (Orlando, 2002) pp. 905–908
47.90 I. McCowan, H. Bourlard: Microphone array post-filter based on noise field coherence, IEEE Trans. Speech Audio Process. **11**(6), 709–716 (2003)
47.91 I. Cohen, S. Gannot, B. Berdugo: An integrated real-time beamforming and postfiltering system for non-stationary noise environments, EURASIP J. Appl. Signal Process. **2003**(11), 1064–1073 (2003), special issue on Signal Processing for Acoustic Communication Systems

47.92 I. Cohen, B. Berdugo: Microphone array post-filtering for non-stationary noise suppression, Proc. ICASSP (Orlando, 2002) pp. 901–904
47.93 I. Cohen, B. Berdugo: Speech enhancement for non-stationary noise environments, Signal Process. **81**(11), 2403–2418 (2001)
47.94 Y. Ephraim, D. Malah: Speech enhancement using a minimum mean-square error short-time spectral amplitude estimator, IEEE Trans. Acoust. Speech Signal Process. **ASSP-32**(6), 1109–1121 (1984)
47.95 Y. Ephraim, D. Malah: Speech enhancement using a minimum mean square error log-spectral amplitude estimator, IEEE Trans. Acoust. Speech Signal Process. **33**(2), 443–445 (1985)
47.96 I. Cohen: Analysis of two-channel generalized sidelobe canceller (GSC) with post-filtering, IEEE Trans. Speech Audio Process. **11**(6), 684–699 (2003)
47.97 S. Gannot, D. Burshtein, E. Weinstein: Theoretical analysis of the general transfer function GSC, Int. Workshop Acoust. Echo Noise Control (IWAENC) (Darmstadt 2001) pp. 27–30
47.98 N. Dal-Degan, C. Prati: Acoustic noise analysis and speech enhancement techniques for mobile radio application, Signal Process. **15**(4), 43–56 (1988)
47.99 S. Gannot, I. Cohen: Audio Sample Files http://www.biu.ac.il/ gannot (2005)

45. Adaptive Echo Cancelation for Voice Signals

M. M. Sondhi

This chapter deals with modern methods of dealing with echoes that may arise during a telephone conversation. Echoes may be generated due to impedance mismatch at various points in a telephone connection, or they may be generated acoustically due to coupling between microphones and loudspeakers placed in the same room or enclosure. If unchecked, such echoes can seriously disrupt a conversation. The device most successful in dealing with such echoes is the adaptive echo canceler. Several million such devices are deployed in the telephone networks around the world. This chapter discusses the principles on which echo cancelers are based, and describes their applications to network telephony and to single- and multichannel teleconferencing.

Echoes are a ubiquitous phenomenon. With rare exceptions, all conversations take place in the presence of echoes. We hear echoes of our speech waves as they are reflected from the floor, walls, and other neighboring objects. If a reflected wave arrives at our ears a very short time after the direct sound, it is perceived not as an echo but as a spectral distortion or reverberation. In general, people prefer some amount of reverberation to a completely anechoic environment. The desirable amount of reverberation depends on the application. (For example, we like more reverberation in a concert hall than in an office.) The situation is very different, however, if a reflected wave arrives a few tens of milliseconds after the direct sound. In this case it is heard as a distinct echo. Such echoes are invariably annoying. Under extreme conditions they can completely disrupt a conversation.

If several reflections arrive in close succession after a long enough delay, the impression is that of a distinct echo with a spectral distortion. It is such, (in general) spectrally distorted, distinct echoes that are the subject of this chapter. Such echoes are of concern in telephone communications because they can be generated *electrically* at points of impedance mismatch along the transmission medium. Such echoes are called *line* or *network* echoes. In a telephone connection between two handsets these are the only type of echoes encountered. In local telephone connections echoes are not a problem because echo levels are low, and if present, they occur after very short delays. However, in a long-distance connection, if the round-trip delay exceeds a few tens of milliseconds, distinct echoes may be heard. The most important source of these echoes is a device called a hybrid, which we will discuss in Sect. 45.1. Hybrids are required whenever a telephone connection uses local networks as well as the long-distance network.

Echoes at hybrids have been a potential source of degradation ever since the advent of long-distance telephony. Many ways of dealing with them have been devised over the past several decades. However, those methods proved to be inadequate for telephone communications via geostationary satellites because of the long round-trip delays – 600 ms or more – encountered on satellite circuits. To deal with echoes arriving with such long delays a new device called an *echo canceler* was developed [45.1–3]. Widespread use of echo cancelers

began around 1980 with the arrival of a very large-scale integration (VLSI) implementation [45.4]. The advent of fiber optics obviated satellite delay. However, more recently, with the growing use of speech coding in the telecommunications network – especially for wireless telephony – delay has again become an issue. At present many millions of echo cancelers are in use in telephone networks around the world.

When a telephone connection is between hands-free telephones (speakerphones) or between two conference rooms, a major source of echoes is the acoustic coupling between loudspeakers and microphones via room reverberation. Adaptive cancelation of such echoes has attracted much attention during the past two decades, and is called *acoustic echo cancelation*. We will discuss this problem in Sect. 45.2. Note, however, that although the echoes are generated acoustically, we will discuss their cancelation only in the *electrical* portion of the circuit. We will not discuss the much more-difficult problem of canceling the echoes *acoustically*, i. e., by generating a canceling acoustic field. Strictly speaking, therefore, we should call our topic the cancelation of acoustically generated echoes. However, the term acoustic echo cancelation is firmly entrenched in the literature.

Finally, in recent years the problem of canceling echoes in stereo (or multichannel) communications has received considerable attention. Although stereophonic teleconferencing has not yet been extensively deployed, we believe that in the not too distant future such conferencing will become quite widespread. Therefore it is important to develop echo cancelers for stereophonic conferencing.

In the next three sections we will discuss network echo cancelers, acoustic echo cancelers, and stereo echo cancelers for voice signals. Note, however, that since their initial introduction for voice signals, echo cancelers were proposed for data signals as well, and have found wide application for such signals. However, the implementation and requirements of those cancelers are very different from those of cancelers for voice signals. We will not discuss echo cancelers for data signals in this chapter.

45.1 Network Echoes

As mentioned above, the main source of network echoes is a device called a hybrid. Figure 45.1 shows a highly simplified diagram showing the placement of hybrids in a typical long-distance telephone connection. Each of the telephones shown is connected to a central office by a two-wire line called the subscriber's loop. For a local call the two subscribers are connected to the same central office. So a local call is set up by simply connecting the two subscribers' loops at the central office. If the distance between the two telephones is long enough (usually, longer than about 35 miles) amplification becomes necessary. Therefore a separate path is needed for each direction of transmission. The device that connects the four-wire part of the circuit to the two-wire portion at each end is known as a hybrid. With reference to Fig. 45.1, the purpose of the hybrids is to allow signals from A to go along the path L_1 to B, and to go from B along the path L_2 to A. However, they must prevent signals in path L_1 from returning as an echo along the path L_2 back to A. Similarly, the signal in path L_2 is to be prevented from returning along path L_1 as an echo back to B.

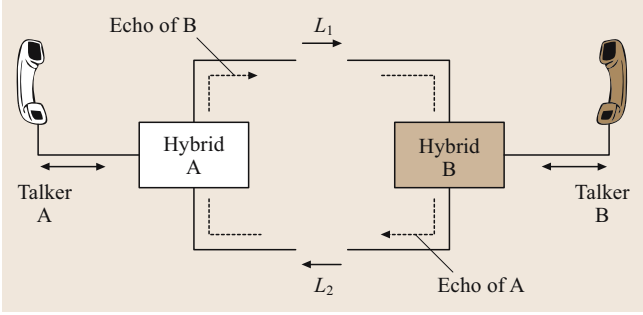


Fig. 45.1 Illustration of a long-distance connection showing local two-wire loops connected through hybrids to a four-wire long-distance network

A hybrid is essentially a Wheatstone bridge network that can achieve the objectives stated above, provided the impedance of the subscriber's loop can be exactly balanced by an impedance located at the hybrid. Unfortunately, this is not possible in practice for several reasons. One reason is that there are far fewer four-wire circuits than there are two-wire circuits. Therefore a hybrid may be connected to any of the subscriber loops served by the central office. By their very nature, subscribers' loops have a wide variety of characteristics – various lengths, type of wire, type of telephone, number of extension phones, etc. Therefore it is impossible to select a single impedance that would balance the

impedance of all subscribers' loops. Besides this, the echo path varies to some extent during a conversation, and can change quite significantly if extension phones are connected and/or disconnected. It appears, therefore, that the echo at the hybrid cannot be completely eliminated by merely selecting an appropriate balancing impedance. As a compromise, a nominal impedance is used to balance the bridge. The average attenuation (in the USA) from input to the return-path output of the hybrid is 11 dB with a standard deviation of 3 dB. The attenuation of the echo provided by the echo path is termed the echo return loss (ERL). An ERL of 9–14 dB is not adequate for satisfactory communication on circuits with long delays [45.5, 6].

For controlling such echoes, a device known as an echo suppressor has long been used – one at each end of the telephone connection. Its principle can be explained by referring to Fig. 45.2, which shows the end B of the telephone circuit of Fig. 45.1, with an echo suppressor included. Based on the level of signals in the paths L_1 and L_2, a decision is made as to whether the signal in L_2 is a signal from B or an echo of A's speech. If the decision is the latter, then the circuit L_2 is opened (or a large loss is switched in). A similar switch at the other end prevents B's echo from returning to B. During intervals when the speech signals from both A and B are simultaneously present at the echo suppressor, suppression is inhibited so that the signal returned to A is an echo of A superimposed on the speech from B. Such occurrence of simultaneous speech from both subscribers is whimsically referred to as *double-talk* in the literature.

If A and B would only talk alternately, it would be possible to design a very good decision mechanism. However, in a telephone conversation it is very important to provide the ability to *break in*, i. e., the ability of B to interrupt when A is talking, and vice versa. Without this ability the conversation would be quite unnatural. If the decision mechanism were to behave flawlessly in spite of break-in, the echo suppressor would be a satisfactory form of echo control. However, when break-in is allowed, the decision cannot be perfect. The two signals that have to be distinguished are both speech signals, with more or less the same statistical properties. Essentially the only distinguishing property is the signal level. For this reason, sometimes too high a level of echo is returned, and sometimes the interrupter's speech is clipped. In spite of these difficulties, echo suppressors have been designed that manage to keep such malfunctions at an acceptable level as long as the round-trip delay is not more than about 60–70 ms. Therefore before the introduction of satellite circuits and/or speech coders, echo suppressors provided a quite satisfactory solution to the echo problem.

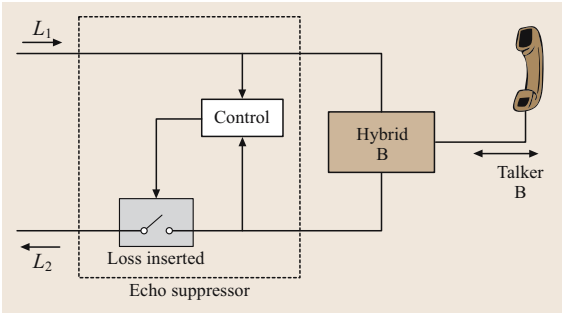


Fig. 45.2 Use of an echo suppressor to remove echo by inserting a switched loss when near-end speech (talker B) is not present

45.1.1 Network Echo Canceler

Echo cancelation is a very different method of echo control that was introduced during the 1960s. Instead of suppressing the echo by introducing a series loss in the circuit, an echo canceler generates a synthetic echo that is a replica of the real echo, and subtracts it from the signal in the echo path. The impetus for the development of echo cancelers came from the fact that echo suppressors failed to provide satisfactory echo control on satellite circuits with their associated long delays. In a circuit via a geostationary communications satellite the round-trip delay of the echo can be about 600 ms. Due to this long delay, the dynamics of a conversation changes, and erroneous switching by the echo suppressors becomes too frequent to be acceptable.

Figure 45.3 shows the same circuit as Fig. 45.2, but with the echo suppressor replaced by an echo

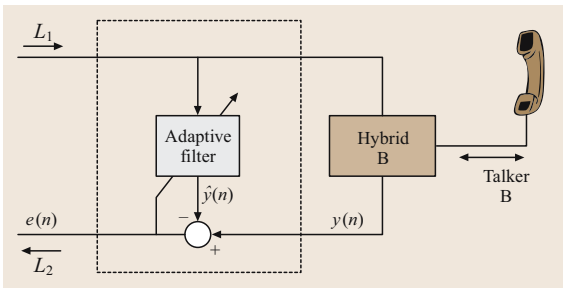


Fig. 45.3 An echo canceler continually removes echo even if the near-end talker is active

canceler. With respect to this echo canceler, the signal from A will be referred to as the far-end signal and the signal from B as the near-end signal. For the canceler located at end A, the roles of the near- and far-end signals are obviously reversed. The function of an echo canceler is to provide additional echo return loss – termed echo return loss enhancement (ERLE) – while allowing uninterrupted communication in both directions.

The box labeled *adaptive filter* in Fig. 45.3 generates a synthetic echo by configuring a filter with a transfer characteristic that matches that of the echo path. For a linear echo path that means matching the impulse response (or, equivalently, the transfer function). Note that the filter has to be *adaptive* in order to cope with the variability of the path due to the reasons mentioned in the previous section. The alternative would be to measure the impulse response periodically by introducing test signals at the input of the echo path. Such intrusions into a telephone conversation are not acceptable. Therefore the filter must adapt to the echo path by using just the existing speech signals. In the next section we will discuss several adaptation algorithms suitable for echo cancelers.

Before turning to the discussion of adaptation algorithms let us mention that a complete echo canceler requires two important features in addition to the adaptation algorithm.

1. During periods of *double-talk* (i. e., periods during which speech from both the near end and the far end are present simultaneously at the canceler) all known adaptation algorithms fail. Therefore an echo canceler must have a double-talk detector that senses this condition. Whenever this condition is detected, the adaptation algorithm is disabled by effectively opening the adaptation path shown in Fig. 45.3. Note that this *does not* interrupt the return path. It only disables adaptation for the (generally quite short) duration of simultaneous talking.
2. The other feature that is required is what is known in the literature as a nonlinear processor. The nonlinearity used is a center clipper. Its function is to set to zero all signals in the return path that are below a certain threshold level. This is necessary because in practice echo cancelers can provide only about 20–25 dB of cancelation. In general a greater degree of cancelation is required, and the center clipper provides that. Since the clipping threshold is very low, the effect of the nonlinear distortion on the near-end signal is not noticeable.

45.1.2 Adaptation Algorithms

The first step in implementing an adaptive filter is to select a structure for the filter such that it can be represented in terms of a finite number of parameters. For linear filters this can be done by modeling the impulse response of the filter as an expansion in terms of any complete set of basis vectors. If $w_l(n)$, $l = 0, 1, 2, \ldots, L-1$, is the set of basis functions, and h_l are the corresponding expansion coefficients, then we assume that the echo $y_e(n)$ is related to the input $x(n)$ by the relation

$$y_e(n) = x(n) * \sum_{l=0}^{L-1} h_l w_l(n) = \sum_{l=0}^{L-1} h_l x_l(n) = \boldsymbol{h}^T \boldsymbol{x}(n) , \quad (45.1)$$

where $*$ indicates convolution $x_l(n) = x(n) * w_l(n)$ and the boldface characters are column vectors of dimension L. The superscript 'T' indicates matrix transpose, and n is the time index. Thus this structure is equivalent to passing the input signal through a bank of L parallel filters with impulse responses $w_l(n)$ and forming a linear combination of the outputs to generate the echo.

The problem thus reduces to the estimation of the expansion coefficient vector $\boldsymbol{h}$. This is done by iteratively adjusting the estimate vector $\hat{\boldsymbol{h}}$ of an adaptive filter so as to drive it towards $\boldsymbol{h}$.

The most popular choice of basis functions is $w_l(n) = \delta(n-l)$, and the resulting filter structure is called a transversal filter (also referred to as a tapped delay line filter or a moving average filter). Its output is just a linear combination of the most recent L samples of the input. For speech signals on the telephone network the sampling frequency $f_s = 8000$ samples per second is used. This is the Nyquist rate for a signal of bandwidth 4 kHz, which is more than adequate for telephone-quality speech whose bandwidth is about 3.4 kHz. At this sampling rate, L is usually chosen to be about 800. For acoustic cancelers, as we shall see, L might have to be chosen to be several thousand.

The adaptation algorithms that we will discuss are strongly dependent on the fact that the output of the adaptive filter is linear in the adapted coefficient vector $\hat{\boldsymbol{h}}$. We will present the adaptation algorithms using the transversal filter as a model. However, except in a few cases, the discussion will be independent of the choice of the component filters w_l. In fact the discussion is valid even if the w_l represent nonlinear filters or filters with infinitely long impulse responses. The actual performance will, of course, depend on the choice of the filter set.

The Stochastic Gradient Algorithm

By far the most popular adaptation algorithm used for echo cancelation is the stochastic gradient algorithm, also popularly known as the least-mean-square (LMS) algorithm. This algorithm was initially introduced around 1960 for adaptive switching [45.7]. It was first used for echo cancelation [45.1] and for adaptive antenna arrays [45.8] in the mid 1960s. Since then it has been used in a variety of applications such as noise cancelation, equalization, etc.

Let the signal $y(n)$ returned from the hybrid be given by

$$y(n) = y_e(n) + v(n) , \tag{45.2}$$

where $y_e(n) = \boldsymbol{h}^T \boldsymbol{x}(n)$ is the echo of the far-end signal $x(n)$, and $v(n)$ is the near-end signal of talker B plus any circuit noise. We will assume that the echo path impulse response $\boldsymbol{h}$ has the form given in (45.1). If that is not strictly true, then the term $v(n)$ will also include the modeling error. Let $\hat{\boldsymbol{h}}(n)$ be an estimate of $\boldsymbol{h}$ at time n and $\hat{y}(n) = \hat{\boldsymbol{h}}^T(n)\boldsymbol{x}(n)$ the corresponding estimate of $y(n)$. We wish to update $\hat{\boldsymbol{h}}(n)$ to an improved vector $\hat{\boldsymbol{h}}(n+1)$ by adding an increment $\boldsymbol{\Delta}$. Since $\boldsymbol{h}$ is unknown we must evaluate the goodness of the estimate indirectly. One measure of the performance of $\hat{\boldsymbol{h}}(n)$ is the error

$$e(n) = y(n) - \hat{y}(n) .$$

Since the objective is to make $\hat{\boldsymbol{h}}$ approximate $\boldsymbol{h}$, one might search for the vector that minimizes the expected value of the squared error $e^2(n)$. One way to do this is to move $\hat{\boldsymbol{h}}$ some distance in the direction of the negative gradient of this expected value. Thus we might choose

$$\boldsymbol{\Delta} = -\frac{\mu}{2} \nabla E[e^2(n)] , \tag{45.3}$$

where E denotes mathematical expectation, μ is the step size parameter that controls the rate of change, and ∇ is the gradient with respect to $\hat{\boldsymbol{h}}(n)$. This update corresponds to the update of the LMS algorithm if the *expected value* of e^2 is replaced by the *instantaneous* value. This seems to be a drastic approximation. However, as we shall see, even such a crude estimate of the gradient is adequate, under certain reasonable conditions, to make $\hat{\boldsymbol{h}}(n)$ approach $\boldsymbol{h}$. Making this approximation and evaluating the gradient in (45.3) yields the LMS update

$$\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(n) + \mu e(n)\boldsymbol{x}(n) . \tag{45.4}$$

Rather than minimizing e^2 one could minimize some other symmetric function of e that has an odd monotonic derivative. That results in a slight generalization of (45.4) to

$$\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(n) + \mu F[e(n)]\boldsymbol{x}(n) , \tag{45.5}$$

where F is any odd monotonic function of its argument. Two choices of the function F have been found useful in practice. The sign function was found useful in early analog implementations of the echo canceler [45.1] because it replaced analog multipliers by switches. However, it slows down convergence and is therefore no longer used. The other function is the ideal limiter with an adaptive threshold, which has been found useful to improve robustness of the adaptation. We will discuss this later.

Before discussing the convergence properties of the algorithm, let us point out that the update of (45.4) has one undesirable property. Observe that both $\boldsymbol{x}$ and e are proportional to the strength of the far-end signal $x(n)$. Thus the size of the update is proportional to the power of the far-end signal. Since the power of a speech signal has a large dynamic range, it follows that the rate at which the algorithm converges would be highly variable. To avoid this variability, the second term on the right-hand side of (45.4) is normalized by dividing it by the L_2 norm $\| \boldsymbol{x}(n) \|^2 = \boldsymbol{x}^T(n)\boldsymbol{x}(n)$, of the vector $\boldsymbol{x}(n)$. This results in the normalized LMS algorithm (NLMS):

$$\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(n) + \frac{\mu}{\| \boldsymbol{x}(n) \|^2} \boldsymbol{x}(n)e(n) . \tag{45.6}$$

We will discuss only this normalized form of the LMS algorithm, since the algorithm without normalization is almost never used. (The appropriate normalization for the algorithm of (45.5) has to be worked separately for each chosen function F.)

As a practical matter, a small positive constant δ is added to the denominator in (45.6), to avoid division by zero. So the final form of the NLMS algorithm is

$$\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(n) + \frac{\mu}{\| \boldsymbol{x}(n) \|^2 + \delta} \boldsymbol{x}(n)e(n) .$$

For the sake of simplicity of notation, we will not show the term δ in the discussion below.

Convergence in the Ideal Case. Consider the ideal case such that

1. the echo path impulse response $\boldsymbol{h}$ is constant, and
2. the error $e(n)$ consists of just the echo of the far-end signal (i. e., the case when the term $v(n)$ is absent in (45.2)).

In this ideal case it is easy to show that the update of (45.6) improves the estimate $\hat{\boldsymbol{h}}$, provided only that $0 <$

$\mu < 2$. To show this let us define the misalignment vector $\boldsymbol{r}(n)$ as the difference between $\boldsymbol{h}$ and $\hat{\boldsymbol{h}}$:

$$\boldsymbol{r}(n) = \boldsymbol{h} - \hat{\boldsymbol{h}}(n) \,. \tag{45.7}$$

In the ideal case

$$e(n) = \boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{r}(n) \,. \tag{45.8}$$

Subtracting $\boldsymbol{h}$ from both sides of (45.6) and rearranging terms we get

$$\boldsymbol{r}(n+1) = \boldsymbol{r}(n) - \frac{\mu}{\| \boldsymbol{x}(n) \|^2}\boldsymbol{x}(n)e(n) \,. \tag{45.9a}$$

Equation (45.9a) can also be written in an alternative useful form as

$$\boldsymbol{r}(n+1) = \left(\boldsymbol{I} - \mu\frac{\boldsymbol{x}(n)\boldsymbol{x}^{\mathrm{T}}(n)}{\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{x}(n)}\right)\boldsymbol{r}(n) \,, \tag{45.9b}$$

where $\boldsymbol{I}$ is the identity matrix.

From (45.9a) it follows that

$$\| \boldsymbol{r}(n+1) \|^2 = \| \boldsymbol{r}(n) \|^2 + \mu(\mu-2)\frac{e^2(n)}{\| \boldsymbol{x}(n) \|^2} \,, \tag{45.10}$$

where we used (45.8) in evaluating the right-hand side. Equation (45.10) shows that, for $0 < \mu < 2$, $\| \boldsymbol{r}(n+1) \|^2 \leq \| \boldsymbol{r}(n) \|^2$, where the equality holds if and only if $e(n) = 0$. Thus as long as there is an uncanceled echo, the length of the misalignment vector decreases, i. e., $\hat{\boldsymbol{h}}$ moves closer to $\boldsymbol{h}$. This conclusion can also be drawn directly from (45.9b) when it is recognized that $(\boldsymbol{x}(n)\boldsymbol{x}^{\mathrm{T}}(n)/ \| \boldsymbol{x}(n) \|^2)\boldsymbol{r}(n)$ is an orthogonal projection of $\boldsymbol{r}(n)$ on $\boldsymbol{x}(n)$. The largest decrease in the length of $\boldsymbol{r}(n)$ occurs for $\mu = 1$. In practice, however, μ is usually set at about 0.3–0.5 to prevent instability due to the perturbations that were neglected in deriving (45.9a) and (45.9b).

Equation (45.10) can be used to show that $e^2(n) \to 0$ as $n \to \infty$. Note that summing the two sides of the equation for $0 \leq n \leq N-1$ and rearranging terms gives

$$\frac{\mu(2-\mu)}{\| \boldsymbol{x}(n) \|^2} \sum_{n=0}^{N-1} e^2(n) = \| \boldsymbol{r}(0) \|^2 - \| \boldsymbol{r}(N) \|^2 \,. \tag{45.11}$$

In view of (45.10), for $0 < \mu < 2$ the right-hand side of (45.11) is bounded, which implies that $e^2(N) \to 0$.

It is not enough, however, to make just the error go to zero. We would in fact like the misalignment vector $\boldsymbol{r}(n)$ to go to zero so that, once the echo canceler has converged, the echo should be canceled for all subsequent inputs. However, without further restrictions on the signal $x(n)$ (45.10) does not allow us to conclude that the misalignment $\boldsymbol{r}(n) \to 0$. This is because $e(n) = 0$ does not imply that $\boldsymbol{r}(n) = 0$, but only that $\boldsymbol{r}(n)$ is orthogonal to $\boldsymbol{x}(n)$. Intuitively speaking, therefore, convergence of $\boldsymbol{r}(n)$ requires that the time-varying vector $\boldsymbol{x}(n)$ is not confined to a subspace of dimension less than L for too long, i. e., $\boldsymbol{x}(n)$ should evolve so as to cover the entire L-dimensional space. This condition is referred to as a mixing condition [45.9] or *persistent excitation* [45.10, pp. 748–751].

Even if $\boldsymbol{x}(n)$ satisfies the mixing condition it is quite difficult to derive accurate estimates of the *rate* of convergence of $\boldsymbol{r}(n)$. Even if $x(n)$ is a member of a stationary ergodic process it is difficult to estimate the expected rate of convergence. Taking the expectation of the two sides of (45.10) gets us nowhere because the right-hand side depends on $\boldsymbol{r}$ itself. However, if μ is sufficiently small, it is reasonable to assume that $\boldsymbol{r}$ changes slowly and the expectation over the $\boldsymbol{x}$ ensemble can be taken assuming $\boldsymbol{r}$ to be quasiconstant [45.1]. This effectively assumes that $\boldsymbol{x}$ and $\boldsymbol{r}$ are statistically independent. Under this independence assumption some useful bounds can be derived for convergence rate when the far-end signal $x(n)$ is a stationary random process. Note that $e^2(n) = \boldsymbol{r}^{\mathrm{T}}(n)\boldsymbol{x}(n)\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{r}(n)$. Therefore with the independence assumption, taking the expectation of (45.10) gives

$$\begin{aligned} &E[\| \boldsymbol{r}(n+1) \|^2] \\ &= E[\| \boldsymbol{r}(n) \|^2] + \mu(\mu-2) \\ &\quad \times E\left[\boldsymbol{r}^{\mathrm{T}}(n)E_x\left[\frac{\boldsymbol{x}(n)\boldsymbol{x}^{\mathrm{T}}(n)}{\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{x}(n)}\right]\boldsymbol{r}(n)\right] \,, \end{aligned} \tag{45.12}$$

where E_x is the expectation over the x ensemble. For a stationary process one may make the approximation that for reasonably large L, $\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{x}(n) \approx L\sigma^2$ where σ^2 is the variance of the x process. With this approximation (45.12) becomes

$$\begin{aligned} &E[\| \boldsymbol{r}(n+1) \|^2] \\ &= E[\| \boldsymbol{r}(n) \|^2] + \frac{\mu(\mu-2)}{L}E\left[\boldsymbol{r}^{\mathrm{T}}(n)\boldsymbol{R}_{xx}\boldsymbol{r}(n)\right] \,, \end{aligned} \tag{45.13}$$

where $\boldsymbol{R}_{xx}$ is the normalized correlation matrix of $\boldsymbol{x}$. Since the second term on the right-hand side of (45.13) can be bounded in terms of the largest and smallest

eigenvalues of the matrix $\boldsymbol{R}_{xx}$, we get

$$\left[1-\frac{\mu(2-\mu)\lambda_{\max}}{L}\right]^n \le \left[\frac{E(\| \boldsymbol{r}(n) \|^2)}{\| \boldsymbol{r}(0) \|^2}\right] \le \left[1-\frac{\mu(2-\mu)\lambda_{\min}}{L}\right]^n . \tag{45.14a}$$

For the range of values $0<\mu<2$, (45.14a) provides geometrically decreasing upper and lower bounds for $E[\| \boldsymbol{r}(n) \|^2]$. With the approximation $(1-x/L)^L \approx \mathrm{e}^{-x}$, the bounds given by (45.14a) are seen to be essentially exponential:

$$\exp\left[-\mu(2-\mu)\frac{n\lambda_{\max}}{L}\right] \le \left[\frac{E(\| \boldsymbol{r}(n) \|^2)}{\| \boldsymbol{r}(0) \|^2}\right] \le \exp\left[-\mu(2-\mu)\frac{n\lambda_{\min}}{L}\right] . \tag{45.14b}$$

In practice the bounds provided by (45.14a, 14b) are surprisingly good. There is little in the literature concerning convergence rates without the independence assumption; only some very loose bounds may be found in [45.9] and [45.11].

Convergence in the Nonideal Case. In practice the convergence process is much more complicated than that discussed above. First of all, the speech signal is a highly nonstationary signal with ill-defined statistical properties. Therefore the bounds on convergence derived above can be used only as a rough guide in designing a canceler. Besides that, there is additive noise and the echo path is not stationary. Very little can be said quantitatively about the effects of these perturbations on the convergence process. Some bounds can be derived [45.9] that show that, in the presence of such perturbations, the algorithm drives the vector $\boldsymbol{r}$ to within a sphere around the origin whose radius is proportional to the root-mean-squared value of the noise and to the rate of change of the impulse response.

The most serious perturbation occurs during periods of *double-talk*, i. e., during intervals when the speech from both the near-end and the far-end speakers is present simultaneously at the echo canceler. The near-end signal is an interference to the adaptation. If the echo canceler has converged to a small misalignment, the near-end signal can be very large compared to the uncanceled echo. In such a situation the adaptation algorithm will quickly misalign the canceler. As mentioned in Sect. 45.1.2, the only known way to deal with this problem is to detect periods of double-talk and disable adaptation during those intervals. The most commonly used double-talk detector is the so-called Geigel algorithm [45.12] that declares double-talk whenever

$$|y(n)| > \alpha \max_{n-L+1\le m\le n} |x(m)| , \tag{45.15}$$

where α is a threshold that is determined by the minimum echo return loss expected from the hybrid (usually set at $\alpha=0.5$, corresponding to a 6 dB hybrid loss). If double-talk is declared, the step-size parameter μ is set to 0. However, the signal e, which now contains the near-end speech along with the residual echo, is returned to the far end. Thus no interruption in the conversation takes place. Whenever double-talk is declared adaptation is interrupted for at least a specified minimum interval so as to avoid very rapid switching of the detector.

This simple algorithm works quite well for network echoes, where minimum hybrid return loss is generally well defined. The selection of α is not so straightforward for acoustic echo cancelers where the echo return loss may even be negative (i. e., the echo may be stronger than the far-end signal). We will discuss strategies employed for those cancelers in Sect. 45.2.5 when we discuss acoustic echo cancelers.

Another Derivation of the NLMS Algorithm. Although the NLMS algorithm is traditionally derived as a gradient algorithm, there is another interesting way of deriving the NLMS update that admits of other generalizations. To show that derivation, let us define an a posteriori error $\varepsilon(n)=y(n)-\hat{\boldsymbol{h}}^{\mathrm{T}}(n+1)\boldsymbol{x}(n)$, i. e., the error that would result if one used the updated estimate of $\hat{\boldsymbol{h}}$ with the current vector $\boldsymbol{x}$.

Obviously, unless all the components of $\boldsymbol{x}(n)$ are zero, it is always possible to find a vector $\hat{\boldsymbol{h}}(n+1)$ in a subspace of dimension $L-1$ such that $\varepsilon(n)=0$ or, more generally, such that $\varepsilon(n)=(1-\mu)e(n)$. If just any arbitrary vector is chosen in that subspace, clearly one learns nothing about the true echo path response $\boldsymbol{h}$. However, with $\hat{\boldsymbol{h}}(n+1)=\boldsymbol{h}(n)+\boldsymbol{\Delta}$, let us find the update $\boldsymbol{\Delta}$ with the smallest norm $\| \boldsymbol{\Delta} \|^2$ that makes $\varepsilon(n)=(1-\mu)e(n)$. Thus let us minimize $(1/2)\| \boldsymbol{\Delta} \|^2$ subject to the constraint $\varepsilon(n)=(1-\mu)e(n)$, or, equivalently the constraint

$$\boldsymbol{\Delta}^{\mathrm{T}}\boldsymbol{x}(n)=\mu e(n) . \tag{45.16}$$

The constrained minimization is equivalent to minimizing

$$\left(\frac{1}{2}\right) \| \boldsymbol{\Delta} \|^2 + \lambda[\boldsymbol{\Delta}^{\mathrm{T}}\boldsymbol{x}(n) - \mu e(n)] \,, \tag{45.17}$$

where λ is a Lagrange multiplier to be determined so as to satisfy the constraint (45.16). Setting the derivative of the expression (45.17) with respect to $\boldsymbol{\Delta}$ to zero gives

$$\boldsymbol{\Delta} = \lambda \boldsymbol{x}(n) \,.$$

Substituting this into (45.16) gives $\lambda = \frac{\mu e(n)}{[\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{x}(n)]}$. Hence

$$\boldsymbol{\Delta} = \frac{\mu}{\| \boldsymbol{x}(n) \|^2} \boldsymbol{x}(n) e(n) \,, \tag{45.18}$$

which is precisely the NLMS update given in (45.6).

The PNLMS Algorithm

Until recently, the NLMS algorithm has been the exclusive choice for network echo cancelers. A new algorithm called the proportionate NLMS (PNLMS) algorithm has been introduced recently [45.13]. It significantly improves convergence rate if the echo path impulse response is *sparse*, i. e., if only a small subset of the L components of $\boldsymbol{h}$ are active, and the rest are zero. [In order to study the effectiveness of algorithms for sparse impulse responses, we need to define sparseness quantitatively. This can be done in several ways. For example, for an impulse response of length L, one may define sparseness S as $S = (L - \sum_{i=0}^{L-1} |h_i|/h_{\max})/(L-1)$. Reference [45.14] gives other possible definitions.] This is precisely the property that makes it useful for network cancelers. As mentioned in Sect. 45.1.2, the length $L$of the transversal filter $\hat{\boldsymbol{h}}$ is usually chosen to be quite large – several hundred taps. This is necessary in order to accommodate the range of impulse responses encountered in the network. However, any particular impulse response is active only at a small (but unknown) subset – a few tens – of the taps.

To motivate the derivation of the PNLMS algorithm, we note that the bounds of (45.14a, 14b) show that the rate at which the NLMS algorithm converges becomes slower as the number of adapted coefficients L increases. Therefore, if it were possible to identify and adapt only the active taps the convergence rate could be significantly improved for sparse echo paths. The PNLMS algorithm does the identification using the following intuitive reasoning. As the NLMS algorithm starts adapting, the coefficients for the inactive taps will stay around zero, while those corresponding to the active taps will build up in magnitude. Therefore the PNLMS algorithm makes the step-size parameter μ proportional to the current estimate of each coefficient. This is accomplished by use of a diagonal matrix $\boldsymbol{G}(n)$, whose diagonal elements $g_{ll}(n)$ are *essentially* the magnitudes of the components of the current vector $\hat{\boldsymbol{h}}(n)$, i. e., $g_{ll}(n) = |\hat{h}_l(n)|, l = 1, 2, \ldots, L$.

In terms of the matrix $\boldsymbol{G}(n)$, the PNLMS update is defined as

$$\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(n) + \frac{\mu}{\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{G}(n)\boldsymbol{x}(n)} \boldsymbol{G}(n)\boldsymbol{x}(n)e(n) \,. \tag{45.19}$$

Since $\boldsymbol{G}(n)$ is diagonal, it follows that (45.19) effectively makes the step-size parameter for each component of $\hat{\boldsymbol{h}}(n)$ proportional to the magnitude of that component. If $\boldsymbol{G}(n)$ is chosen to be the identity matrix, of course, the algorithm reduces to the NLMS algorithm.

As in the case of the NLMS algorithm, a small constant δ is added to the denominator to avoid division by zero. Also, as a practical matter, the diagonal entries of $\boldsymbol{G}(n)$ must be modified somewhat from those given above, to avoid some singular behavior. Note, for instance, that if at any instant n a diagonal entry of $\boldsymbol{G}(n)$ becomes zero the corresponding coefficient will stop adapting. Indeed, if the vector $\hat{\boldsymbol{h}}(n) = 0$, (as is usually the case initially, at $n = 0$) no adaptation would take place. Also, components that are very small should still be adapted at some minimum rate. Finally, the average value of the diagonal elements should be held constant in order to be able to select a value of μ for stable operation. If the average value is set to unity, then μ can be chosen such that $0 < \mu < 2$ as for NLMS. To take these factors into account, the diagonal elements of $\boldsymbol{G}(n)$ are modified as

$$\begin{aligned}
\gamma_{\min} &= \rho \max\left\{\delta_p, |\hat{h}_1(n)|, |\hat{h}_2(n)|, \ldots, |\hat{h}_L(n)|\right\} \,, \\
\gamma_l(n) &= \max\left\{\gamma_{\min}, |\hat{h}_l(n)|\right\} \,, \quad 1 \le l \le L \,, \\
g_{ll}(n) &= \frac{\gamma_l(n)}{\frac{1}{L}\sum_{l=1}^{L} \gamma_l(n)} \,.
\end{aligned} \tag{45.20}$$

The parameter δ_p, typically chosen to be 0.01, prevents stalling if the vector $\hat{\boldsymbol{h}}(n) = 0$. The parameter $\gamma_{\min}$ provides a minimum adaptation rate for each coefficient. The parameter ρ (typically chosen to be about $5/L$) sets the value of $\gamma_{\min}$ relative to the largest adapted coefficient. Finally, the third line of (45.20) makes the average value of the diagonal elements equal to unity.

The PNLMS algorithm has been extensively tested and shown to be significantly better than NLMS for network cancelers. And, as can be seen, it requires only

a small increase in the computational requirements compared to NLMS ($3L$ multiplies compared to $2L$). It has been incorporated in recently developed network echo cancelers.

It is interesting to note that, although the PNLMS algorithm has been used successfully on telephone networks, no satisfactory proof of its convergence is known at present. Let us show the difficulty that arises in an attempt at a convergence proof.

Consider only the ideal noiseless situation with a perfectly constant echo path. As in the case of the NLMS algorithm, subtract both sides of (45.19) from the true response vector $\boldsymbol{h}$ and rewrite the equation in terms of the misalignment vector

$$\boldsymbol{r}(n+1) = \boldsymbol{r}(n) - \frac{\mu}{\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{G}(n)\boldsymbol{x}(n)}\boldsymbol{G}(n)\boldsymbol{x}(n)\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{r}(n)\,. \tag{45.21}$$

We will assume that $\boldsymbol{G}(n)$ is symmetric and positive definite. [The specification of (45.20) assures that. The following discussion is valid for any positive definite symmetric matrix.] Then we can define the positive-definite symmetric square-root matrix $\boldsymbol{G}^{1/2}(n)$, such that $\boldsymbol{G}^{1/2}(n)\boldsymbol{G}^{1/2}(n) = \boldsymbol{G}(n)$.

In terms of $\boldsymbol{G}^{1/2}(n)$ let us define the vectors $\boldsymbol{s}(n) = \boldsymbol{G}^{-1/2}(n)\boldsymbol{r}(n)$, $\boldsymbol{s}(n+1) = \boldsymbol{G}^{-1/2}(n+1)\boldsymbol{r}(n+1)$, and $\boldsymbol{u}(n) = \boldsymbol{G}^{1/2}(n)\boldsymbol{x}(n)$. Substituting these into (45.20) and rearranging terms we get

$$\boldsymbol{G}^{-1/2}(n)\boldsymbol{G}^{-1/2}(n+1)\boldsymbol{s}(n+1) = \boldsymbol{s}(n) - \frac{\mu}{\boldsymbol{u}^{\mathrm{T}}(n)\boldsymbol{u}(n)}\boldsymbol{u}(n)\boldsymbol{u}^{\mathrm{T}}(n)\boldsymbol{s}(n)\,. \tag{45.22}$$

If the matrix $\boldsymbol{G}$ were a constant independent of n, then the product of the two matrices on the left-hand side would be an identity and (45.22) would be identical to the NLMS update of (45.9b) with $\boldsymbol{x}$ replaced by $\boldsymbol{u}$ and $\boldsymbol{r}$ replaced by $\boldsymbol{s}$. In this case the discussion of the convergence properties of NLMS would carry over unchanged to the convergence of $\boldsymbol{s}(n)$, except that the covariance matrix of $\boldsymbol{x}$ would be replaced by the covariance matrix of $\boldsymbol{u}$, $\boldsymbol{R}_{uu} = \boldsymbol{G}^{1/2}\boldsymbol{R}_{xx}\boldsymbol{G}^{1/2}$. We could therefore conclude that $\| \boldsymbol{s}(n) \| \to 0$, and hence $\| \boldsymbol{r}(n) \| \to 0$ (since $\boldsymbol{G}^{1/2}$ is positive definite). However, since $\boldsymbol{G}$ is not constant, we cannot draw this conclusion. If the parameter μ is small, one can argue that $\boldsymbol{G}^{-1/2}(n)\boldsymbol{G}^{1/2}(n+1) \approx \boldsymbol{I}$, in which case convergence can again be inferred. It is also possible to get some exponential bounds on the convergence process (as for NLMS) in terms of the eigenvalues of $\boldsymbol{G}^{-1/2}(n)\boldsymbol{G}^{1/2}(n+1)$. However there does not appear to be any simple way to relate those eigenvalues to the properties of the far-end signal $x(n)$. Clearly, the convergence properties of the PNLMS algorithm are not yet well understood.

In spite of this lack of understanding of its convergence properties, intuitive reasoning has been used to provide an improvement of the PNLMS algorithm. The main drawback of the PNLMS algorithm is that although it performs much better than the NLMS algorithm for sparse impulse responses, it does not perform well (indeed it is even worse than NLMS) for impulse responses that are not sparse. In the telephone network nonsparse impulse responses are rare but they do occur, so it is desirable to be able to deal with both sparse and nonsparse responses. One simple solution (dubbed PNLMS++) is to alternate PNLMS and NLMS updates. A much better solution is the improved PNLMS (IPNLMS) algorithm proposed in [45.15]. This algorithm uses an update that is a compromise between the NLMS and the PNLMS updates. The update is accomplished by replacing the diagonal matrix $\boldsymbol{G}$ defined in (45.20) by a diagonal matrix $\boldsymbol{K}$ whose diagonal elements k_{ll} are obtained by replacing the corresponding elements g_{ll} of $\boldsymbol{G}$ as

$$k_{ll}(n) = \frac{1-\alpha}{2L} + (1+\alpha)\frac{|\hat{h}_l(n)|}{2\sum_{l=1}^{L}|\hat{h}_l(n)|}\,, \tag{45.23}$$

where α is a parameter such that $-1 \le \alpha < 1$. For $\alpha = -1$, k_{ll} is constant independent of l, so IPNLMS is identical to NLMS. For α close to 1, k_{ll} is essentially proportional to $|h_l(n)|$, so IPNLMS behaves like PNLMS. In [45.15] it is shown that, for α close to 0 or -0.5, IPNLMS always behaves better than both NLMS and PNLMS.

We close the discussion of the PNLMS algorithm by deriving it via a constrained minimization quite similar to the one used in deriving the NLMS algorithm. Again, let $\hat{\boldsymbol{h}}(n+1) = \boldsymbol{h}(n) + \boldsymbol{\Delta}$, and require that the a posteriori error be related to $e(n)$ as $\varepsilon(n) = (1-\mu)e(n)$, which is the same as requiring that

$$\boldsymbol{\Delta}^{\mathrm{T}}\boldsymbol{x}(n) = \mu e(n)\,.$$

Again we find the update with the minimum norm that satisfies this constraint. Only instead of the L_2 norm we choose the norm $\boldsymbol{\Delta}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Delta}$, where $\boldsymbol{M}$ is some, as yet unspecified, symmetric positive definite matrix. Thus we need to minimize

$$\frac{1}{2}\boldsymbol{\Delta}^{\mathrm{T}}\boldsymbol{M}^{-1}\boldsymbol{\Delta} + \lambda[\boldsymbol{\Delta}^{\mathrm{T}}\boldsymbol{x}(n) - \mu e(n)]\,.$$

Setting the derivative of this expression with respect to $\boldsymbol{\Delta}$ equal to zero gives

$$\boldsymbol{\Delta} = \lambda \boldsymbol{M}\boldsymbol{x}(n) \, .$$

Substituting this in the constraint equation gives

$$\lambda = \frac{\mu e(n)}{\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{M}\boldsymbol{x}(n)} \, ,$$

and hence

$$\boldsymbol{\Delta} = \frac{\mu}{\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{M}\boldsymbol{x}(n)} \boldsymbol{M}\boldsymbol{x}(n)e(n) \, .$$

If $\boldsymbol{M}$ is chosen to be the matrix $\boldsymbol{G}(n)$ as defined by (45.20) we get the PNLMS update. If $\boldsymbol{M}$ is chosen to be the matrix $\boldsymbol{K}$ defined by (45.23) we get the IPNLMS update.

More generally, choosing different specifications for the matrix $\boldsymbol{M}$ yields different updates. For instance $\boldsymbol{M} = \boldsymbol{G}^2(n)$ is a reasonable choice. That would measure the norm of the vector of relative increments (i. e., the vector with components $(\Delta_l/\hat{h}_l)$. Reference [45.14] considers other possibilities for the matrix $\boldsymbol{M}$ that are suitable for sparse impulse responses (e.g., the exponential gradient algorithm). To date, however, no choice has been found that provides any useful improvement over IPNLMS for network echo cancelation.

Let us note in passing that the form of the PNLMS update, as well as our derivation of it, has a similarity to the natural gradient update [45.16]. The natural gradient update just takes a step in the direction of $\boldsymbol{\Delta}$. This is equivalent to setting $\lambda = \mu e(n)$. Thus PNLMS is just a normalized form of the natural gradient update, and has been discussed from that point of view in [45.17].

Robust LMS and PNLMS

As mentioned in the Section on the ideal case, a double-talk detector is essential to prevent the LMS algorithm from diverging in the presence of double-talk. However, a double-talk detector [e.g., the Geigel detector of (45.15)] cannot turn off adaptation instantaneously, and even a very short interval of undetected double-talk can significantly increase the misalignment. A slight modification of the LMS algorithm can make it quite robust to such a short-lived disturbance [45.18]. The modification consists of replacing $e(n)$ in the NLMS update equation by $F[e(n)]$ as shown in (45.5). Likewise, for the PNLMS algorithm the same change is made in its update (45.19). From the discussion following (45.5) it would appear that the choice of a fixed function F in general slows down convergence and has therefore not been found to be useful. However, robustness can be achieved if the function is adapted. The form of the function is chosen such that it can be adjusted depending on the presence or absence of the disturbance. As noted in [45.18], $F(e)$ must be chosen such that $\lim_{|e|\to\infty} |F(e)| < \infty$, i. e., F has to be a saturating nonlinearity. A simple and convenient choice is $F(e) = \mathrm{sign}(e)sH\,(|e|/s)$, where H is a hard limiter, i. e.,

$$H\left(\frac{|e|}{s}\right) = \frac{|e|}{s} \, , \quad \frac{|e|}{s} \le k$$
$$= k \, , \quad \frac{|e|}{s} > k \, . \tag{45.24}$$

In (45.24), s is a positive scale factor whose effect depends upon the manner in which it is adapted. Initially, we would like s to be large so that H is in the linear region. Then $F(e) = e$ and the updates are just the normal NLMS updates. As the canceler converges we would like s to follow the level of the uncanceled echo to lower values ending with a value somewhat larger than the expected level of the circuit noise at the near end. During a short burst of double-talk (for the short interval required for the double-talk detector to act) we would like s to stay small and let the temporarily large value of e drive H into the saturation region, thus limiting the convergence rate. A simple adaptation of s that achieves this objective is given by

$$s(n+1) = \lambda_{\mathrm{s}} s(n) + (1-\lambda_{\mathrm{s}})\frac{1}{\beta} s(n) H\left(\frac{|e(n)|}{s(n)}\right) \, , \tag{45.25}$$

where λ_{s} is an exponential forgetting factor and β is a constant with a value of about 0.7.

The initial value $s(0)$ is set to be somewhat larger than the expected level of speech signals on the telephone network, so that initially H is in the linear range. As long as H stays in the linear range, $s(n)$ is $1/\beta$ times the average value of $|e(n)|$ in the immediate past. The forgetting factor λ_{s} gives an averaging interval of roughly $1/(1-\lambda_{\mathrm{s}})$ samples, so λ_{s} is chosen at some value between 0.99 and 1. As the canceler converges, the scale factor decreases from its initial value to a value slightly larger than the level of the circuit noise. At the initiation of double-talk, for a duration of roughly $1/(1-\lambda_{\mathrm{s}})$ samples, the scale factor stays low, the nonlinearity H saturates, and the rate of divergence is reduced. This gives time for the double-talk detector to act and prevent further divergence. This robustness does not come for free. If the error increases suddenly due to a sudden change in the echo path, the canceler will not track the change right away. However, as the scale factor estimator adapts to a larger value, the convergence rate

accelerates. There is thus a trade-off between immunity to double-talk and the rate of recovery from a sudden change of echo path.

Other Algorithms

For network echo cancelation, as far as we know, no other algorithms have found use.

For acoustic echo cancelation some other algorithms have been found useful.

An algorithm known as the two-path model was proposed in the 1970s, and has some interesting properties. However, since economy of computational requirements was the overriding criterion at the time, and the improvement it afforded for network applications was not significant, it did not gain favor. Recently there has been a revived interest in it for use in acoustic echo cancelation. Another algorithm that has been useful for acoustic echo cancelation is the subband canceler, as well as frequency-domain and multidelay filters. These algorithms turn out to be useful for the acoustic case because of the much longer impulse responses encountered in that application. We will discuss these in Sect. 45.2.

Adaptive filtering, however, has an extensive literature [45.10]. Several other algorithms selected from that literature have been proposed for echo cancelation, but have not yet found wide application. We will mention some of them briefly here, but will not discuss them in any detail.

Another Variable-Step Algorithm. PNLMS is a variable-step algorithm in that it assigns different adaptation step sizes to the different coefficients of the estimated response vector $\hat{\boldsymbol{h}}(n)$. It provides fast convergence for sparse echo paths. However, there is one undesirable property of PNLMS that deserves attention. Suppose the algorithm has converged to a small misalignment. According to the algorithm, the coefficient $\hat{h}_j(n)$ will be assigned a step size essentially proportional to the magnitude $|h_j|$ of the corresponding coefficient of the echo path response. This means that the step size will be large for the large coefficients, even though they might have converged to their correct values. Obviously, this is not a desirable assignment when the coefficients have converged. In that case all coefficients should have small step sizes to keep the residual misalignment low. Ideally one would like the step size to be proportional to $|\hat{h}_j(n) - h_j|$. This, of course, is not possible because h_j is not known. However, an algorithm described in [45.19] assigns step sizes in a way that makes them smaller as the estimated coefficient comes close to its target. The algorithm is based on the simple observation that, when a coefficient has converged close to its target, the successive updates of that coefficient will no longer push it consistently in one direction. So for each coefficient, the sign of the update is monitored over D consecutive samples in the immediate past. If the number of sign changes exceeds a specified number N_1, the step size is decreased; if it is less than a specified number N_2, the step size is increased. The parameters D, N_1, and N_2, and the range over which the step size is allowed to vary, are determined empirically. This algorithm was proposed as a general adaptive filter algorithm. As far as we know, it has not been applied to echo cancelation. Intuitively speaking, it seems to have the right properties for assigning step sizes near convergence. In the initial phase, however, the step assignment according to PNLMS should give faster convergence. Some combination of the two should yield better performance than that of either one by itself.

The Block Least-Squares Algorithm. In the section on the stochastic gradient algorithm we noted that $\hat{\boldsymbol{h}}$ can be estimated by minimizing the expected value of the squared error $e^2(n)$. However, for simplicity, we took the instantaneous value of the squared error as an estimate of its expected value. An improvement in performance can be expected if better estimates of the expected error are used. One such estimate is the average, or arithmetic mean, of $e^2(m)$ over a range of values of the index m. At time index n, let us average over the range $n-M+1 \le m \le n$. Ignoring the normalizing constant M, we see that $\hat{\boldsymbol{h}}(n)$ may be estimated as the vector $\hat{\boldsymbol{h}}$ that minimizes the quantity

$$
\begin{aligned}
J(\hat{\boldsymbol{h}}) &= \sum_{m=n-M+1}^{n} e^2(m) \\
&= \sum_{m=n-M+1}^{n} [y(m) - \hat{\boldsymbol{h}}^{\mathrm{T}} \boldsymbol{x}(m)]^2 \\
&= \hat{\boldsymbol{h}}^{\mathrm{T}} \sum_{m=n-M+1}^{n} \boldsymbol{x}(m)\boldsymbol{x}^{\mathrm{T}}(m)\hat{\boldsymbol{h}} - 2\hat{\boldsymbol{h}}^{\mathrm{T}}\boldsymbol{w}(n) \\
&\quad + \sum_{m=n-M+1}^{n} y^2(m) \\
&= \hat{\boldsymbol{h}}^{\mathrm{T}} \boldsymbol{X}(n)\boldsymbol{X}^{\mathrm{T}}(n)\hat{\boldsymbol{h}} - 2\hat{\boldsymbol{h}}^{\mathrm{T}}\boldsymbol{w}(n) + \sum_{m=n-M+1}^{n} y^2(m)\,.
\end{aligned}
\tag{45.26}
$$

In the last line of (45.26) $\boldsymbol{X}(n)$ is an $L \times M$ matrix whose m-th column is $\boldsymbol{x}(n-M+m)$ for $1 \le m \le M$, and the

vector $\boldsymbol{w}(n)$ is given by $\boldsymbol{w}(n) = \sum_{m=n-M+1}^{n} y(m)\boldsymbol{x}(m)$. Setting the gradient of J with respect to $\hat{\boldsymbol{h}}$ equal to zero shows that the minimum is obtained as the solution to

$$\boldsymbol{X}(n)\boldsymbol{X}^{\mathrm{T}}(n)\hat{\boldsymbol{h}}(n) = \boldsymbol{w}(n)\,. \tag{45.27}$$

For the solution to be unique, the matrix $\boldsymbol{X}(n)\boldsymbol{X}^{\mathrm{T}}(n)$ must be of full rank. If the matrix is not of full rank (either because $M < L$, or because the number of independent rows is $< L$), a pseudoinverse can be used to yield a minimum norm solution.

The solution of (45.27) requires the inversion of an $L \times L$ matrix, which would normally require $O(L^3)$ operations. However, due to the special Toeplitz structure of $\boldsymbol{X}$, the inversion can be done with approximately $12L^2$ operations [45.20].

The solution depends on the time index n as well as on the averaging interval M. Given the nonstationary nature of the problem, $\hat{\boldsymbol{h}}(n)$ should be updated as often as the available computational resources allow. Note, however, that even if the inversion is done once every block, the per-sample computations are approximately $12L$. That is still several times more than the computations required for NLMS.

The Recursive Least-Squares Algorithm. In the block least-squares algorithm, the estimate of $\hat{\boldsymbol{h}}(n)$ is obtained once per block. The recursive least-squares (RLS) algorithm allows an update of the algorithm every sample. In this algorithm, instead of approximating the expected value of $e^2(n)$ by the average over a block, it is approximated by a weighted sum of the squared error at all past samples. With an exponential weighting, one minimizes

$$J(\hat{\boldsymbol{h}}) = \sum_{m=-\infty}^{n} \lambda^{n-m} e^2(m)\,, \tag{45.28}$$

where the *forgetting factor* λ is chosen in the range $0 < \lambda < 1$. The solution proceeds as for the case of the block algorithm, except that equation to be solved is

$$\boldsymbol{R}(n)\hat{\boldsymbol{h}}(n) = \boldsymbol{w}(n)\,, \tag{45.29}$$

with $\boldsymbol{R}(n) = \sum_{m=-\infty 1}^{n} \lambda^{n-m}\boldsymbol{x}(m)\boldsymbol{x}^{\mathrm{T}}(m)$ and $\boldsymbol{w}(n) = \sum_{m=-\infty}^{n} y(m)\boldsymbol{x}(m)$. From these definitions the following recursions are seen to be valid:

$$\boldsymbol{R}(n) = \lambda\boldsymbol{R}(n-1) + \boldsymbol{x}(n)\boldsymbol{x}^{\mathrm{T}}(n) \tag{45.30}$$

and

$$\boldsymbol{w}(n) = \lambda\boldsymbol{w}(n-1) + y(n)\boldsymbol{x}(n)\,. \tag{45.31}$$

Because of the recursion (45.30), the matrix inversion lemma [45.9, p. 480] can be used to compute $\boldsymbol{R}^{-1}(n)$ efficiently by updating $\boldsymbol{R}^{-1}(n-1)$. Thus the estimate of $\hat{\boldsymbol{h}}(n)$ can be obtained recursively from $\hat{\boldsymbol{h}}(n-1)$. The procedure is straightforward, but a bit cumbersome. The details may be found in [45.10, Chap. 13]. The computational requirement of the recursion is of order $O(L^2)$ per iteration. By making use of the shift property of the transversal filter the computational requirements can be reduced to as low as $7L$ per iteration The algorithms that achieve this low rate are the fast recursive least-squares (FRLS) and fast transversal filter (FTF) algorithms [45.21–23]. Their drawback is that they are rather numerically unstable, and have to be periodically reset.

The RLS algorithm generally gives much faster convergence than NLMS. However, its tracking ability is not necessarily better [45.24].

Affine Projection Algorithms. The affine projection algorithm (APA) is a generalization of the NLMS algorithm. It was first introduced in 1984 [45.25]. By definition, an affine projection is the combination of a projection and a translation. Let us write the NLMS update of (45.6) in a slightly different form using (45.7) and (45.8):

$$\begin{aligned}\hat{\boldsymbol{h}}(n+1) &= \hat{\boldsymbol{h}}(n) + \boldsymbol{\Delta}\\ \boldsymbol{\Delta} &= -\mu\boldsymbol{x}(n)[\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{x}(n)]^{-1}\\ &\quad\times\boldsymbol{x}^{\mathrm{T}}(n)[\hat{\boldsymbol{h}}(n) - \boldsymbol{h}(n)]\,.\end{aligned} \tag{45.32}$$

The matrix $\boldsymbol{x}(n)[\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{x}(n)]^{-1}\boldsymbol{x}^{\mathrm{T}}(n)$ is a projection operator that orthogonally projects $[\hat{\boldsymbol{h}}(n) - \boldsymbol{h}(n)]$ on $\boldsymbol{x}(n)$. Therefore $\boldsymbol{x}(n)[\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{x}(n)]^{-1}\boldsymbol{x}^{\mathrm{T}}(n)[\hat{\boldsymbol{h}}(n) - \boldsymbol{h}(n)]$ is an affine projection of $\hat{\boldsymbol{h}}(n)$ on $\boldsymbol{x}(n)$. Thus the NLMS update is obtained by subtracting from $\hat{\boldsymbol{h}}(n)$, μ times an affine projection of $\hat{\boldsymbol{h}}(n)$ on $\boldsymbol{x}(n)$. Instead of subtracting the projection of $[\hat{\boldsymbol{h}}(n) - \boldsymbol{h}(n)]$ on a single vector, the APA subtracts the projection on the subspace spanned by the past M vectors $\boldsymbol{x}(m)$, $n - M + 1 \le m \le n$. This is accomplished by simply replacing $\boldsymbol{x}(n)$ in (45.32) by the $L \times M$ matrix whose columns are these M vectors. That is in fact the matrix $\boldsymbol{X}(n)$ defined just before (45.27). Thus the APA is obtained by replacing $\boldsymbol{x}(n)$ by $\boldsymbol{X}(n)$ in (45.32), i. e.,

$$\begin{aligned}\hat{\boldsymbol{h}}(n+1) &= \hat{\boldsymbol{h}}(n) + \boldsymbol{\Delta}\\ \boldsymbol{\Delta} &= -\mu\boldsymbol{X}(n)[\boldsymbol{X}^{\mathrm{T}}(n)\boldsymbol{X}(n)]^{-1}\\ &\quad\times\boldsymbol{X}^{\mathrm{T}}(n)[\hat{\boldsymbol{h}}(n) - \boldsymbol{h}(n)]\,.\end{aligned} \tag{45.33}$$

If the columns of $\boldsymbol{X}(n)$ are independent, then as M increases they span a larger subspace of the L-dimensional

space of the $\boldsymbol{x}(n)$ and $\hat{\boldsymbol{h}}(n)$ vectors. Therefore, in general, the convergence rate increases significantly as M increases.

The price paid for the faster convergence is higher computational cost. The APA requires $2LM + KM^2$ multiplications per iteration, where K can be as low as 7 [45.26]. If $M \ll L$ then the algorithm is essentially M times as costly as NLMS. If the update is computed only once every M samples the algorithm is called the partial-rank algorithm (PRA), which was introduced originally in [45.27] for adaptive beam forming. The cost per sample of the PRA is comparable to that of NLMS, but its convergence is slower than that of the APA. With highly colored signals such as speech the convergence rate of PRA is more or less the same as that of APA. The main disadvantage of PRA compared to APA is that its computations are very unevenly spread out over an M-sample cycle. To overcome this disadvantage a fast affine projection (FAP) algorithm that requires $2L + 30M$ computations per iteration has been developed [45.26, 28]. One disadvantage of the FAP algorithm is that, although the current error signal is saved, the current estimate of $\hat{\boldsymbol{h}}$ is not saved. What is saved instead is a vector that depends on both the current estimate of $\hat{\boldsymbol{h}}$ as well as the current matrix $\boldsymbol{X}$. For network echo cancelers, this can lead to undesirable performance. Note that during periods of double-talk and during silence at the far end, adaptation is inhibited. Since network echo paths usually change very slowly, NLMS needs to reconverge only to a slightly changed echo path. An algorithm that does not save the estimate of the echo path must re-converge from scratch. For acoustic echo cancelers this might not be such a detriment because the echo path is highly variable, and hence NLMS would also have to reconverge to a significantly changed echo path.

We turn next to acoustic echo cancelers.

45.2 Single-Channel Acoustic Echo Cancelation

Acoustic echoes arise in hands-free telephony and in teleconferencing due to the coupling between the loudspeaker and the microphone via the transfer characteristics of the room in which they are located. A typical configuration is illustrated in Fig. 45.4. In that figure we have shown one end of a teleconference where a four-wire connection is set up between the two conference rooms. If the connection is a two-wire connection then hybrids must be included in the circuit. We neglect them for the present discussion.

In principle the problem of acoustic echoes is not fundamentally different from the problem of network echoes. Instead of the echo being generated due to impedance mismatch, it is now generated due to the acoustic coupling between the loudspeaker and the microphone. Other than that the problem in the two cases is identical. However, the techniques required to deal with acoustic echoes are significantly different from those required for network echoes. The main difference is that the properties of the echo paths in the two situations are very different [45.29, 30].

One main difference is that the room impulse response is much longer than the response of a network connection. Recall that in a network connection the impulse response of the echo path is usually about 500–600 taps long (at the standard sampling frequency of 8000 samples per second), of which only about 50–60 are active taps. In contrast, a room response even for a small office can be significant over an interval of 300–500 ms, corresponding to 2500–4000 taps, all of which are active.

A second difference is that a room impulse response is much more time varying than a network impulse response. The transmission characteristics of the network vary very slowly (except occasional rapid changes due, for example, to someone picking up or hanging up an extension phone). In contrast, the impulse response of a room is constantly changing. Note that the impulse re-

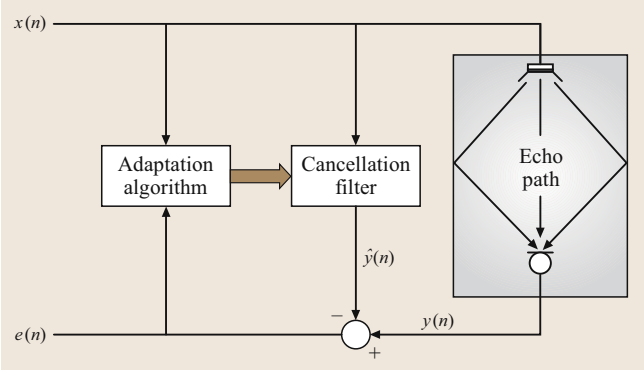


Fig. 45.4 An acoustic echo canceler is used to cancel echoes that arise from coupling between a loudspeaker and microphone

sponse of a room is made up of reflections from the walls, ceiling, floor, and all objects and people in the room. As people move around, the impulse response changes. Opening or closing of doors or windows changes the echo path. Even changes in the speed of sound due to temperature changes can produce a significant amount of misalignment. (A temperature change of 5.5 °C produces roughly a 1% change in the speed of sound. This corresponds to a change of about one sampling interval, i. e., 125 μs, in travel time for an acoustic path of about 4 m.)

Thus we note that acoustic echo cancelers require more computing power due to the increased length of the impulse response to be identified and also due to the faster convergence required to track the faster time variation. We will discuss some of the techniques used to deal with these problems.

45.2.1 The Subband Canceler

The technique used for dealing with the increased length of the impulse response is to utilize subband echo cancelers. The structure of such a canceler is shown in Fig. 45.5. It was first proposed in 1984, independently in [45.31] and [45.32] and further developed in [45.33]. The idea is to use a bank of contiguous bandpass filters with impulse responses $w_0, w_1, w_2, \ldots, w_{K-1}$ that span the frequency range of the speech signal. Rather than taking a linear combination of the outputs of these filters as the model for the echo (as discussed in Sect. 45.1.2), the output of each bandpass filter is used as an input to a transversal filter. The coefficients of each transversal filter are adapted using an NLMS algorithm so that it cancels the echo in its subband. (The symbol $\Rightarrow$ is used in Fig. 45.5 to indicate a bundle of paths, one for each subband filter.) Thus the structure is equivalent to M echo cancelers – one for each subband. The outputs of subband filters, after cancelation of the respective echoes, are summed to give the final signal returned to the far end.

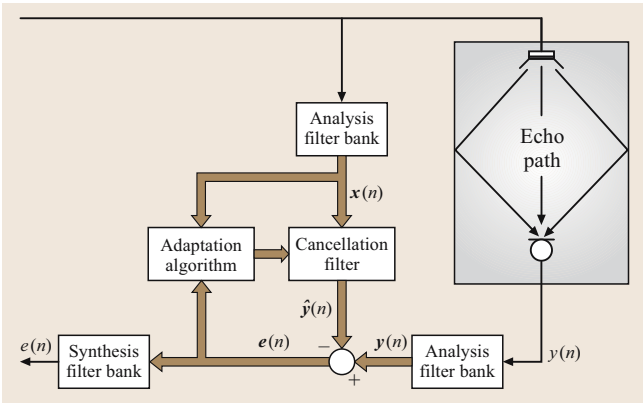


Fig. 45.5 Block diagram of a subband echo canceler showing analysis and synthesis filter banks, subband cancelation filter, and adaptation algorithm

The following simple calculation shows that the number of computations per second required for this structure is $1/M$ times that required for a single NLMS algorithm for the entire impulse response.

Let the room response have duration T_H seconds. Let L be the number of samples representing this impulse response. Let the sampling interval for the full-band speech signal be T_s seconds. (As mentioned above, the normal sampling frequency f_s on the telephone network is 8000 samples per second, which gives $T_s = 125\,\mu s$.) Assume that the impulse response when bandlimited to each subband has the same duration T_H as the full-band impulse response. If M subbands are used, the minimum required sampling frequency for each subband signal is f_s/M. Hence each subband impulse response has L/M coefficients. Each of these has to be adapted once in MT_s seconds. Thus each subband requires $L/(M^2T_s)$ adaptations per second. For all the M subbands combined, the total number of adaptations is therefore $L/(MT_s)$ per second, which is M times smaller than the L/T_s adaptations per second required for a single full-band canceler.

In this derivation we have neglected the overhead required for the analysis and synthesis filters shown in Fig. 45.5. If this overhead is taken into account, the subband structure provides computational savings only if T_H is longer than about 90 ms. Thus for network cancelers the subband structure is not very attractive. However, for room responses it provides a significant reduction in computational requirements.

Clearly, the reduction in computational complexity derived above depends on sampling the subband signals at a rate $1/M$ that of the full-band signal. This requires the use of multirate filters described in [45.33, 34]. As shown in [45.33], exact cancelation can occur only if the filters are free of aliasing. In general, less aliasing requires longer analysis and synthesis filters, with consequent additional transmission delay. Design of these filters is thus a compromise between computational complexity, transmission delay, and the degree of aliasing. These aspects of the design of subband filters are discussed in some detail in [45.35].

It should be noted that the sampling frequency of f_s/M for the subband filters is the minimum sampling

rate (also called critical sampling). Sampling at this critical frequency makes the control of aliasing very difficult. In practice, therefore, the sampling frequency is chosen to be f_s/R with R somewhat smaller than M. With this change, the computations are reduced by the somewhat smaller factor R^2/M. The reduction factor approaches M as R approaches the critical sampling rate M.

The performance of a given design depends on the interplay of two sources of error: the aliasing introduced by the analysis filters and the error due to the finite length of the transversal filters in each subband. If very short transversal filters are used, the truncation error dominates. Increasing the length of the transversal filters allows smaller misalignment. However, once the aliasing error becomes the controlling factor, further increase in length does not afford any significant improvement.

The subband structure has several advantages. The main advantage, as mentioned earlier, is the very significant reduction in computational complexity. This reduction in computational complexity can be exploited in several ways: hardware can be saved to reduce cost, the overall system bandwidth or the duration of the impulse response to be modeled can be increased, or more-complex adaptation algorithms can be employed. An example of the use of a more-complex algorithm is given in Sect. 45.2.2. Another advantage is that the structure of the subband canceler is well suited to parallel processing. A third advantage is that it provides an improved convergence rate. The reason for this is that the speech spectrum has a much smaller dynamic range within any one of the subbands. Therefore, the eigenvalues of the correlation matrix for the signal within each subband have smaller spread. Some small eigenvalues are introduced at the edges of each subband, due to the transition band of the filter response. These small eigenvalues can in fact *increase* the eigenvalues spread. However, the analysis and synthesis filters reduce the error signal at those frequencies, so the effect of these eigenvalues on the overall misalignment is small. Initially, therefore, the subband canceler converges faster than a full-band canceler. The small eigenvalues do cause slow asymptotic convergence [45.36]. One possible solution to this problem is to make the analysis filter bandwidth slightly larger than the pass band of the synthesis filter. This was proposed in [45.36] and implemented in [45.37]. One final advantage of the subband structure is in the design of the nonlinear processor that is used to eliminate the residual echo. The center clipper can now be placed in each subband. This produces much less distortion of the near-end signal at the same clipping level. Alternatively, it allows one to increase the threshold for the same overall distortion of the near-end signal, thus allowing a less-expensive canceler with a larger misalignment.

45.2.2 RLS for Subband Echo Cancelers

If economy of computation is not the overriding consideration, each subband canceler can be adapted by algorithms other than NLMS. The most ambitious such modification for acoustic echo cancelation to date is the algorithm proposed in [45.38], which uses the subband structure but instead of the NLMS algorithm it uses the FRLS algorithm. As mentioned, FRLS tends to be numerically unstable, and has to be frequently restarted. That introduces frequent undesirable interruptions. However, in the subband structure, it is possible to restart the algorithm in one band at a time, in some sequence that periodically visits all the subbands. When there are a large number of subbands, the slight short-lived misalignment in any one band is perceptually unnoticed.

45.2.3 The Delayless Subband Structure

The only *disadvantage* of the subband approach is the extra delay introduced by the analysis and synthesis filters. This delay depends on the design of the analysis and synthesis filters, but can be 20 ms or more (double that for the round-trip delay). For many applications

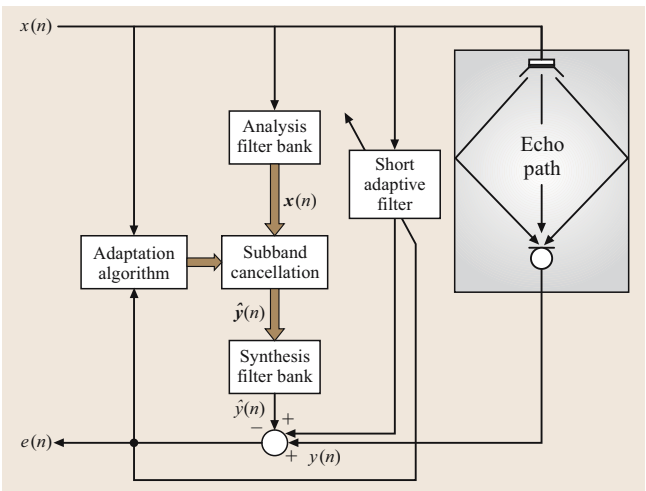


Fig. 45.6 Delayless subband adaptive filter. An auxiliary short wide-band adaptive filter is needed to fill the gap due to the delay in the analysis and synthesis filters

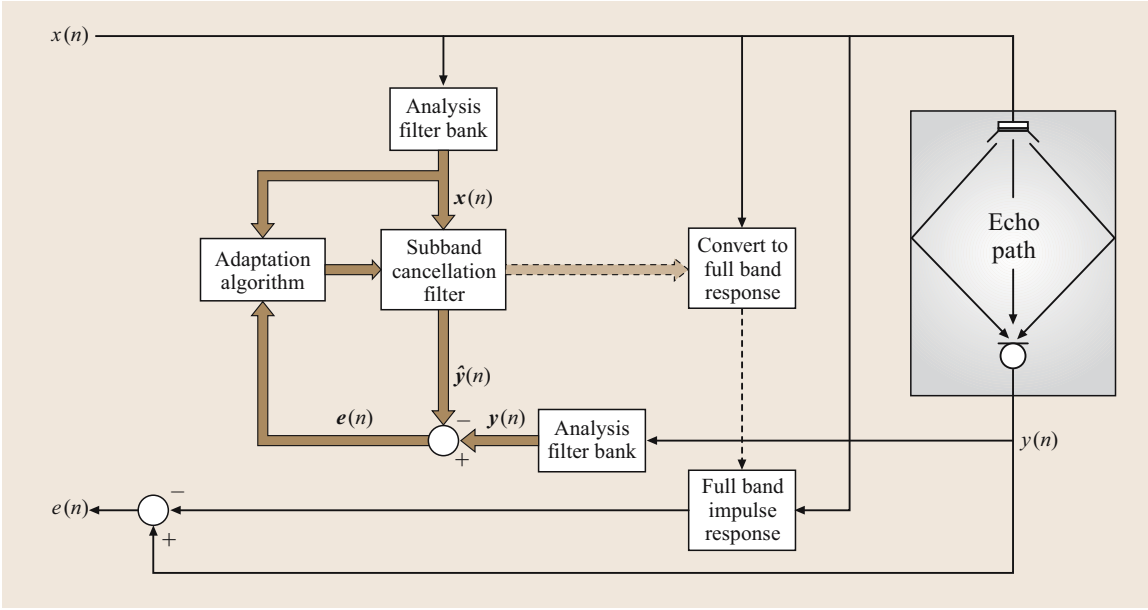


Fig. 45.7 The delayless subband structure of [45.39] that eliminates signal path delay by transforming weights of the canceling subband filter to the equivalent wide-band filter

this extra delay is undesirable. Two methods have been suggested to avoid this delay.

One proposal [45.40] eliminates the delay by moving the analysis and synthesis filters from the signal return path to the cancelation filter path as shown in Fig. 45.6. When this is done, the canceler cannot model the initial part of the echo impulse response due to the delay introduced in the model path. Therefore, a short full-band canceler is added in parallel to model the initial portion of the echo path response. This structure has two drawbacks. Delay in a feedback loop is always problematic, especially when the impulse response is time-varying. Also, the extra computations needed for the auxiliary full-band canceler, reduces the computational advantage of the subband structure.

The second proposal [45.39] is to use the subband structure for adaptation as above but transform the weights to the equivalent single full-band transversal filter to do the actual cancelation of the echo in the return signal path. This structure is shown in Fig. 45.7. It eliminates the delay in the return path without introducing delay in the feedback loop and without requiring an auxiliary canceler. It does require additional computations for the derivation of the full-band filter, and for the extra full-band convolution. The latter can be done quite efficiently by using a well-known technique: transform the two signals to be convolved to the frequency domain using a fast Fourier transform (FFT), multiply the two transformed signals, and compute the inverse FFT of the product to return to the time domain.

45.2.4 Frequency-Domain Adaptation

Another technique that is used to deal with the long impulse responses of rooms is frequency-domain adaptation. A block diagram of a typical implementation is shown in Fig. 45.8. It uses block processing in which one block of input data is processed at a time and produces one block of output. Referring to Fig. 45.8, the Fourier transform $\tilde{\boldsymbol{x}}$ of a block of samples of the far-end signal $x(n)$ is computed via the fast Fourier transform (FFT). The components of the adapted coefficient vector $\tilde{\boldsymbol{h}}$ now multiply the corresponding components of $\tilde{\boldsymbol{x}}$ to produce the Fourier transform of the output of the filter. (The inverse FFT of $\tilde{\boldsymbol{h}}$ is the coefficient vector of the corresponding transversal filter.) The inverse FFT of the filter output gives the block of time-domain samples of the estimated response. The error between this block and the block of samples of the echo path output y, gives the block of error samples. Its Fourier transform controls the vector $\tilde{\boldsymbol{h}}$ to drive the error towards zero. The coefficient vector $\tilde{\boldsymbol{h}}$ is adapted once per block. The main advantage of this technique is that, because of the use of the FFT, the number of computations per

sample is significantly reduced compared to the NLMS algorithm.

The first such proposal was presented in [45.41]. However, in that proposal circular convolution was used rather than the linear convolution required for modeling the echo response. (Circular convolution is convolution of signals obtained by periodic extension of the signals within a block.) Due to this, the adaptive filter does not converge to the true transversal solution [45.42]. How this drawback can be removed through the use of five FFTs per block is shown in [45.42]. A further improvement is presented in [45.43], where the algorithm requires only three FFTs per block. In these papers in order to realize linear convolution, the transversal filter is extended with a block of zero coefficients, and a two-to-one overlap between successive input blocks is used. The FFT is computed over a two-block window.

In references [45.41,42] and [45.43] the block length used is equal to the length L of the transversal filter. For large L, since the FFT of a window of $2L$ samples requires $2L\log_2(L)$ computations, the number of computations per block is essentially $CL\log_2(L)$, where the constant of proportionality C depends on the particular implementation, but is close to 10. Hence the number of computations per sample is $C\log_2(L)$. The NLMS algorithm requires $2L$ computations per sample, which is substantially greater.

Besides the advantage of reduction in the computational requirements, the frequency-domain approach has two additional advantages. For signals like speech, the components of the Fourier transform tend to be more uncorrelated than successive samples in the transversal filter. Also, the step sizes for the coefficients can be adjusted individually to reduce the disparity in the convergence rates of the components due to the large spread in the eigenvalues of the correlation matrix of the speech signal. One simple way of doing this is given in [45.43].

These good features are offset by two important disadvantages. First, the block processing introduces a delay of L samples, which for a room response is intolerably long. And second, since the filter coefficients are held constant during each block, the ability of the adaptive filter to track changes in the room response is reduced.

The problem of long delay can be alleviated by a proposal in [45.44] called multidelay filter (MDF) adaptation. By sectioning the transversal filter into K nonoverlapping segments the delay is reduced by a factor of K. A generalized MDF is presented in [45.45]. It allows the overlap between successive blocks to be greater than 2 to 1, thus allowing more-frequent adaptation and hence better tracking ability. A unified treatment of these ideas is presented in [45.46, Chap. 8].

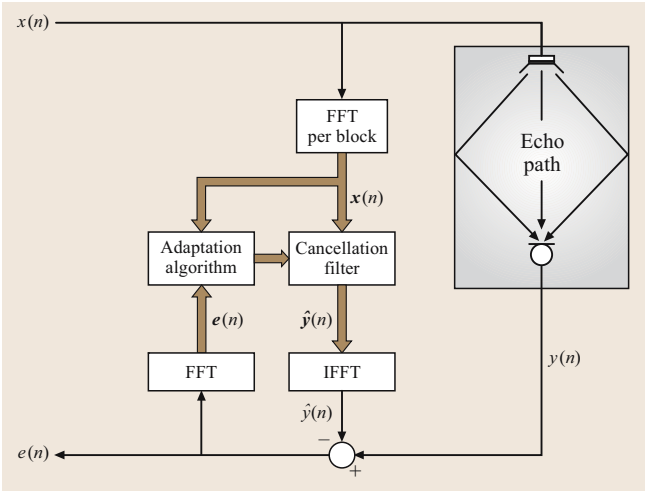


Fig. 45.8 Frequency-domain adaptation. The input is processed in blocks. One output block is generated for each input block

45.2.5 The Two-Echo-Path Model

As mentioned in the section on other algorithms, the two-path model [45.47] was proposed in the 1970s but did not find application for network echo cancelation. It has attracted renewed interest for acoustic echo cancelation.

The basic idea of the two-path model is very simple. Observe that in the implementation of the echo canceler as shown in Fig. 45.3 a single filter serves the dual purpose of adapting itself to the echo path and of canceling the echo in the signal returned to the far end. This provides computational economy but is not necessary. The two functions can be performed by separate filters. This simple observation leads to the idea of the

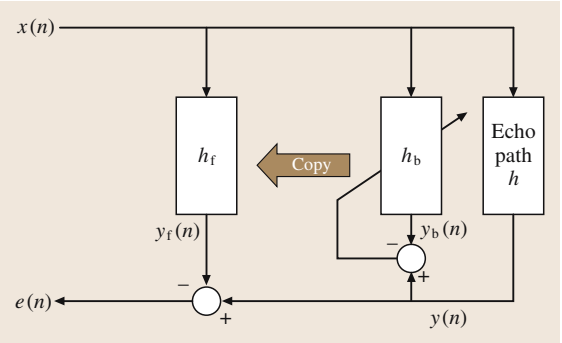


Fig. 45.9 Two-path model of echo canceler

two path model whose structure is shown in Fig. 45.9. A *background* filter adapts as in a conventional echo canceler. A *foreground* canceler does the actual cancelation. Its coefficients are updated by copying from the background canceler whenever a decision logic declares that the background canceler is performing better than the current foreground canceler.

The main reason this method has found favor for acoustic echo cancelation is the fact that it deals very effectively with double-talk in this application. Unlike network echoes, for which the hybrid provides a minimum guaranteed echo return loss, acoustically generated echoes can produce a signal at the microphone whose amplitude is as large as, or often larger than that of the far-end signal. This makes it impossible to set the threshold for the Geigel double-talk detector. Other methods are needed to disable adaptation during double-talk. Several methods have been proposed recently [45.46, Chap. 6]. However, with the two-path model a double-talk detector is not necessary. Any reasonable logic will decide that the background canceler is worse than the foreground canceler when double-talk sets in. Therefore, the foreground canceler is not updated until the double-talk ceases and the background canceler has adapted to a better estimate.

Referring to Fig. 45.9, we define $\hat{\boldsymbol{h}}(n)$ and $\hat{\boldsymbol{h}}_\mathrm{b}(n)$ as the current estimated impulse responses of the foreground and background cancelers, respectively, and $e(n)$ and $e_\mathrm{b}(n)$ the corresponding errors. The decision to update the foreground canceler is, in general, a function of $e(n)$, $e_\mathrm{b}(n)$, the far-end signal $x(n)$, and the echo path signal $y(n)$. The success of this method depends crucially upon the choice of that function. The algorithm for this decision described in [45.47] is not suitable for acoustic echo cancelers. The main reason is that it relies on the hybrid to provide a minimum echo return loss. The algorithm proposed recently in [45.48] has been shown to work very well. It essentially monitors the echo return loss enhancement (ERLE) of the two paths averaged by a first-order integrator with a time constant of about 150 ms. The coefficients of $\hat{\boldsymbol{h}}_\mathrm{b}(n)$ are copied into $\hat{\boldsymbol{h}}(n)$ whenever the average ERLE of the background canceler is better than that of the foreground canceler.

45.2.6 Variable-Step Algorithm for Acoustic Echo Cancelers

During our discussion of the convergence of the PNLMS algorithm, we had mentioned the possibility of using a fixed (i. e., constant over time) matrix $\boldsymbol{G}$ to assign different step sizes to different components of the adaptive filter response vector $\hat{\boldsymbol{h}}$. Indeed, convergence of the PNLMS algorithm was proved only for this special case. This possibility of using a fixed $\boldsymbol{G}$ was already mentioned as early as the 1960s [45.1]. The idea was that the step size for the coefficient $\hat{h}_j$ should be made proportional to the expected value of its target h_j over the ensemble of echo responses on the telephone network. On average, therefore, all coefficients would then require the same number of steps to reach their respective targets. The ensemble of impulse responses on the network turned out to be too varied (especially due to the wide range of initial delays) to make this idea useful. The situation is different in the case of acoustic echo cancelers used for teleconferencing. For any given room the echo response has a fairly well-defined envelope, which is in general exponential, with a decay constant that depends on the reverberation time of the room. The envelope changes when people move around or if the loudspeaker and microphone positions are changed. However, a compromise exponential envelope can be estimated from several measurements in the room. It is therefore reasonable to make the step sizes decay exponentially with coefficient number, with the decay constant of the exponential chosen on the basis of the reverberation time of the room. This idea was presented in [45.49] where it was shown that for a room of the size of a typical office, the technique can make the initial convergence of the NLMS algorithm two to three times faster than with a fixed step size.

45.2.7 Cancelers for Nonlinear Echo Paths

Throughout the discussion so far, we have assumed that the echo path is linear and can therefore be characterized by an impulse response. The problem of echo cancelation for a nonlinear path was considered as early as 1971 [45.50].The proposal was to use the Volterra kernel expansion of the output $y(n)$ of the nonlinear echo path, when its input is the far-end signal $x(n)$. The Volterra expansion has the form

$$y(n) = \sum_{l=0}^{L-1} h_l x(n-l) + \sum_{l_1=0}^{L-1} \sum_{l_2>l_1}^{L-1} h_{l_1 l_2} x(n-l_1)x(n-l_2) + \cdots , \tag{45.34}$$

where the series continues with third- and higher-order terms. The adaptation algorithm then has to estimate

the coefficients $h_l, h_{l_1 l_2}, \ldots$ Note that $y(n)$ depends linearly on these coefficients. Hence the stochastic gradient algorithm can again be used to estimate them.

The major drawback of this method is that the number of coefficients to be adapted becomes enormously large even if third- and higher-order terms are neglected. Fortunately, the telephone network is quite well represented by a linear system; hence the use of Volterra kernels was never required.

For acoustic echo cancelation there has been a revival of some interest in the Volterra kernel expansion. The reason for this interest is that, in certain applications such as echo cancelation in cars, inexpensive loudspeakers are used, and to get sufficient volume they are driven into their nonlinear region. The echo path, which includes the loudspeaker, thus becomes nonlinear. (It may be argued that one could replace the signal $x(n)$ in the adaptation algorithm by the signal picked up by a microphone close to the loudspeaker. However, in general this is a bad idea. The signal from such a microphone would include the echo as well as any near-end speech. Adaptation with such a signal would be problematic, especially in a small space such as a car, where these extraneous disturbing signals can be quite large.)

Recently, the idea of using the Volterra kernel expansion has been proposed to deal with this problem [45.51]. If the Volterra expansion of (45.34) were to be used, the number of coefficients to be estimated would, of course, be prohibitively large. However, one can make use of the fact that the echo path consists of a zero-memory nonlinearity (or a nonlinearity with a very short memory) cascaded with a long linear impulse response. When this structure is taken into account, the number of coefficients can be reduced to a manageable size [45.51].

This concludes our discussion of single-channel acoustic echo cancelation. We turn next to the most modern application of echo cancelers – multichannel acoustic echo cancelers.

45.3 Multichannel Acoustic Echo Cancelation

The most common use of acoustic echo cancelers today is in single-channel communication. Recently, however, there has been a growing interest in stereophonic (or, more generally, multichannel) echo cancelation. The need for such cancelers arises in several applications where the echo picked up by a microphone is due to two or more different, but correlated signals.

The primary motivation for developing stereophonic cancelers is the desire for high-quality teleconferencing between a group of people in one room with another group at a different location. At present such teleconferencing is almost exclusively monophonic. However, it is our belief that in the future many, if not all, teleconferences will be stereophonic. A stereophonic system has several advantages when compared to a monophonic system. With a stereo system each group gets a much better feeling of *presence* at the remote location. Imagine, for instance, half of a circular table around which several people are seated. The table is placed adjacent to a large video screen on which the remote conference room is projected, with several people seated at the *other* half of the table, thereby creating the impression of a circular table with people seated around it. If a monophonic system is used for such a teleconference it would be very difficult to associate voices at the remote location with their respective talkers. With a multichannel system (or at a minimum a stereo system) the task is much easier because sounds can be made to appear to come from the directions of their sources. Not only does a multichannel system provide a much more natural interaction, it also improves intelligibility, especially when two or more people happen to be talking simultaneously. This is due to the well-known *cocktail party* effect – the ability of humans to pay attention to one speaker at a cocktail party, even when several other conversations are taking place simultaneously. The mechanisms by which humans are able to do this are not well understood. However, the ability to identify the directions from which sounds are received is a prerequisite for the cocktail party effect. If one listens to a monophonic recording of a cocktail party, it is very difficult to pick out one conversation.

Besides such teleconferencing, there are several other potential applications for stereophonic echo cancelers. One example is a so called desktop teleconference in which the conferees are all at different locations. Each conferee has a desktop with pictures, or videos of all the other conferees displayed on the screen. As in the conference described in the previous paragraph, the interaction between the conferees is improved if a participant's voice appears to come from the direction of his or her picture (or video). A minimum of two loudspeakers and one microphone are needed at each location to realize this. Another application might arise if one uses

a microphone in a car to communicate with, say, a speech recognizer. For a natural interaction, one would like to be able to do so while the stereo system in the car is in use. This requires a stereo canceler to cancel the signals from the stereo system. Finally, multichannel cancelers are needed for interactive games involving multichannel sound.

At first glance, stereo echo cancelation appears to be just a minor modification of single-channel cancelation. However going from a single channel to two channels produces a basic change in the requirements of an echo canceler. Extension to more than two channels does not introduce any new problems. Hence we will discuss only stereo cancelers here. The extension of the ideas to a system with three or more channels is fairly straightforward.

In order to understand the fundamental problem of stereo acoustic echo cancelation [45.52], let us first describe the configuration of a stereo teleconference. Figure 45.10 shows a schematic of such a conference. As before, we will consider echo cancelation for one end of the circuit (room B), it being understood that analogous configurations are used for echo cancelation in room A. For a discussion of echo cancelation in room B, we will refer to room A as the transmitting room and room B as the receiving room. Each room has two loudspeakers and two microphones. The microphones in the transmitting room pick up signals $x_1(n)$ and $x_2(n)$. These two signals are transmitted to loudspeakers in the receiving room. Each microphone in the receiving room picks up an echo of *both* signals, linearly combined via the acoustic echo paths to the microphone from the two loudspeakers. To keep the discussion simple we will consider the cancelation of the echo picked up by only one of the microphones, as shown in Fig. 45.10, and denote the impulse responses from the loudspeakers to that microphone by $h_1(n)$ and $h_2(n)$. A similar discussion applies to cancelation of the echo at the other microphone, with $h_1(n)$ and $h_2(n)$ replaced by the responses appropriate to that microphone. (For convenience, whenever we refer to the impulse response of an acoustic path we will assume that it includes the responses of the loudspeaker and microphone in that path.)

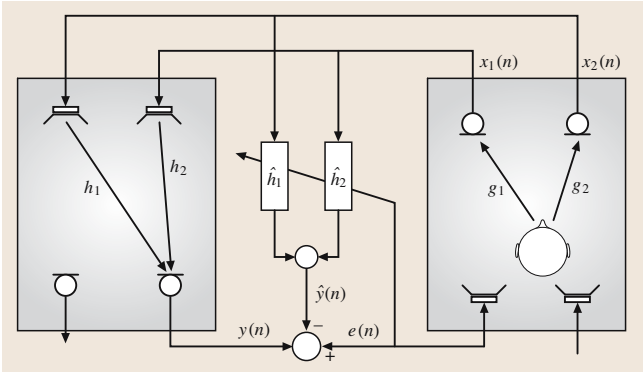


Fig. 45.10 Schematic diagram of stereophonic echo cancelation, showing the use of adaptive filters $\hat{h}_1$ and $\hat{h}_2$ to cancel the echo $y(n)$ arising from the echo paths h_1 and h_2 from the two loudspeakers on the left to one of the microphones

Neglecting ambient noise and signals generated in the receiving room, the signal picked up by the microphone is

$$y(n) = \boldsymbol{h}_1^{\mathrm{T}}\boldsymbol{x}_1(n) + \boldsymbol{h}_2^{\mathrm{T}}\boldsymbol{x}_2(n)\,, \tag{45.35}$$

where we have defined the vectors $\boldsymbol{h}_1$, $\boldsymbol{h}_2$, $\boldsymbol{x}_1$, and $\boldsymbol{x}_2$ analogously to the impulse response and far-end signal vectors described in the single-channel case. In principle the vectors $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$ can have different dimensions. However, without loss of generality we will assume that they have an equal number of components, L, each. The vectors $\boldsymbol{x}_1(n)$ and $\boldsymbol{x}_2(n)$ are then also of the same dimension L. As a straightforward generalization of the single-channel case we now implement two adaptive filters whose outputs are added together to generate a replica of the echo. Denoting the current estimated impulse response vectors of the two adaptive filters as $\hat{\boldsymbol{h}}_1(n)$ and $\hat{\boldsymbol{h}}_2(n)$, we get

$$\hat{y}(n) = \hat{\boldsymbol{h}}_1^{\mathrm{T}}(n)\boldsymbol{x}_1(n) + \hat{\boldsymbol{h}}_2^{\mathrm{T}}(n)\boldsymbol{x}_2(n)\,, \tag{45.36}$$

and the error signal

$$e(n) = y(n) - \hat{y}(n) = \boldsymbol{r}_1^{\mathrm{T}}(n)\boldsymbol{x}_1(n) + \boldsymbol{r}_2^{\mathrm{T}}(n)\boldsymbol{x}_2(n)\,, \tag{45.37}$$

where $\boldsymbol{r}_1(n)$ and $\boldsymbol{r}_2(n)$ are the current misalignment vectors for the two channels. Let us define the *stacked* vector $\boldsymbol{x}(n)$ as the concatenation of $\boldsymbol{x}_1(n)$ and $\boldsymbol{x}_2(n)$ (i. e., $\boldsymbol{x}(n) = [\boldsymbol{x}_1^{\mathrm{T}}(n)\boldsymbol{x}_2^{\mathrm{T}}(n)]^{\mathrm{T}}$). Similarly define the stacked vectors $\boldsymbol{h}$, $\hat{\boldsymbol{h}}(n)$, and $\boldsymbol{r}(n)$. In terms of these vectors the expressions for $y(n)$, $\hat{y}(n)$, and $e(n)$ are exactly the same as the ones in the section on the stochastic gradient algorithm. Therefore one would expect to be able to use the NLMS algorithm of (45.6) in the ideal case to drive the misalignment vector $\boldsymbol{r}(n) \to 0$, i. e., $\hat{\boldsymbol{h}}_1(n) \to \boldsymbol{h}_1$ and $\hat{\boldsymbol{h}}_2(n) \to \boldsymbol{h}_2$. This turns out not to be true. In the ideal case, the NLMS algorithm does drive the error to zero. However, the misalignment does not necessarily converge to zero. This is because, as shown in the next subsection, the condition of zero error can be achieved by infinitely many nonzero vectors $\boldsymbol{r}(n)$. Thus, although

$\boldsymbol{r}(n) = 0$ implies $e(n) = 0$, the converse is not true. The condition $e(n) = 0$ only implies a relationship between the impulse responses of the receiving room and the transmission room. This nonuniqueness is the basic difference between single- and two-channel cancelers. It arises when only one source is active in the transmitting room, and is due to the fact that in such a case the signals $x_1(n)$ and $x_2(n)$ are highly correlated.

45.3.1 Nonuniqueness of the Misalignment Vector

The nonuniqueness of the misalignment vector is a property of the stereo configuration, and is independent of the particular adaptation algorithm used. Thus, ignoring for the moment the question of *how* convergence is achieved, let us just assume that the error has been driven to be identically equal to zero. We will show that the setting of the adaptive filters that achieves this is not unique.

Let us consider the nonuniqueness problem in terms of the signals and impulse responses as time series. An equivalent vector–matrix treatment is, of course, possible, but is somewhat more cumbersome.

The error $e(n)$ can be written

$$e(n) = r_1(n) * x_1(n) + r_2(n) * x_2(n) \,, \tag{45.38}$$

where, as before, $*$ denotes convolution, and $r_1(n)$ and $r_2(n)$ are the misalignments in the two channels. The nonuniqueness problem arises when a single source is active in the transmitting room, as shown in Fig. 45.10. In such a case, the source signal $s(n)$ is convolved with the two transmitting room impulse responses $g_1(n)$ and $g_2(n)$ to generate the two signals $x_1(n)$ and $x_2(n)$. That is $x_1(n) = s(n) * g_1(n)$ and $x_2(n) = s(n) * g_2(n)$. The simplest way to show the nonuniqueness is to note that, since convolution is commutable,

$$g_2(n) * x_1(n) = g_2(n) * g_1(n) * s(n) = g_1(n) * x_2(n) \,. \tag{45.39}$$

In view of (45.39), the expression for the error in (45.38) can be rewritten

$$\begin{aligned} e(n) = {} & [r_1(n) + \alpha g_2(n)] * x_1(n) \\ & + [r_2(n) - \alpha g_1(n)] * x_2(n) \,, \end{aligned} \tag{45.40}$$

where α is an arbitrary constant. Clearly the settings of the adaptive filters $\hat{h}_1(n)$ and $\hat{h}_2(n)$ that achieve zero error are not unique.

Another way to show the nonuniqueness is to rewrite the error in (45.38) as

$$e(n) = [r_1(n) * g_1(n) + r_2(n) * g_2(n)] * s(n) \,. \tag{45.41}$$

For the error to be identically zero for all $s(n)$, the expression in square brackets must be zero, i. e.,

$$r_1(n) * g_1(n) + r_2(n) * g_2(n) = 0 \,. \tag{45.42}$$

In the z-transform domain (45.42) is

$$R_1(z)G_1(z) + R_2(z)G_2(z) = 0 \,, \tag{45.43}$$

where the upper-case quantities denote the z-transforms of the corresponding lower-case quantities. Clearly, (45.42) and (45.43) *do not* imply that the misalignment in each channel is zero. Only the expression involving the two misalignments has to be zero.

The problem with stereo cancelers is evident from (45.43) [or equivalently from (45.42)]. Even if the receiving room impulse responses are constant, any change in G_1 and/or G_2 requires a readjustment of $r_1(n)$ and $r_2(n)$, i. e., of the adaptive filters $\hat{h}_1(n)$ and $\hat{h}_2(n)$. Thus the adaptation algorithm must track not only the variations in the receiving room responses, but also the variations in the transmitting room responses. Given the long impulse responses in a room, it is barely possible to track the changes in just the receiving room alone. To track changes in *both* rooms is quite difficult. Furthermore, the transmitting room responses are particularly difficult to track. If, for instance, one talker stops and another starts speaking the responses $g_1(n)$ and $g_2(n)$ change quite abruptly and by large amounts. The difficult challenge therefore, is to devise an algorithm whose convergence (as in the case of a single-channel canceler) is unaffected by changes in the transmitting room.

Before we discuss currently available solutions to the nonuniqueness problem, let us make two observations concerning stereo cancelation. First, the nonuniqueness problem essentially disappears if two or more statistically independent and spatially separated sources are simultaneously active in the transmitting room. This is because R_1 and R_2 would have to satisfy an equation of the type of (45.43) for each additional independent source, with impulse responses appropriate to that source. Except in some highly improbable situation these equations will be linearly independent. Therefore the only possible solution would be $R_1(z) = R_2(z) = 0$. However, the fact that the solution is unique when two or more sources are active is not of much use for a teleconference because most of the time only one person speaks at a time.

The second observation is that all room responses have tails extending beyond the length L modeled by the adaptive filter. Therefore, theoretically there is the possibility that the frequency response of the transmitting room impulse responses varies more rapidly with frequency than is possible for the adaptive filter response to replicate. This too would tend to force the misalignment to zero in order to satisfy (45.43). However, in any practical situation this fact is also of little consequence because, in any usable implementation of an echo canceler, L has to be large enough that the tails are quite negligible. We will mention the effect of these tails a bit more quantitatively in the following section.

45.3.2 Solutions for the Nonuniqueness Problem

Examination of (45.38) suggests a possible solution, first proposed in [45.52]. If the signals $x_1(n)$ and $x_2(n)$ were uncorrelated they would not be able to compensate for each other. The minimum of the error (zero in the ideal case) would then be obtained only when the two misalignments $r_1(n)$ and $r_2(n)$ were both zero.

Thus, one could try decorrelating the two-channel signals. Of course, $x_1(n)$ and $x_2(n)$ cannot be completely decorrelated because then the stereo effect itself would be lost. The challenge, therefore, is to decorrelate them by an amount adequate to make the adaptive algorithm converge, yet small enough to be perceptually negligible.

Several attempts have been made to exploit this idea. One simple proposal is to add uncorrelated noise signals to $x_1(n)$ and $x_2(n)$. Adding white Gaussian noise does improve convergence [45.52] but it needs to be at a level that degrades the perceptual quality of the signals. Adding so-called Schroeder noise is still not satisfactory. [Schroeder noise for a signal $x(n)$ is given by $\varepsilon\gamma(n)x(n)$, where $\gamma(n)$ are independent identically distributed random variables that take on the values ± 1 with equal probability, and the constant ε controls the level of the noise. This noise has the property that it is white and has a time-varying variance that is proportional to the local variance of $x(n)$. Therefore it is perceptually less disturbing than white noise of constant variance.]

Another method of decorrelation is to introduce a time-varying filter in one of the channels, say $x_1(n)$. One suggestion [45.53] is to periodically introduce a one-sample delay. To get a rough idea of the effect of a one-sample delay, let us make the approximation $x_1(n-1) \approx x_1(n) - Tx_1'(n)$, where T is the sampling interval and x_1' denotes the derivative of x_1. With this approximation it is seen that the proposal is essentially equivalent to periodically adding a noise proportional to the derivative of the signal. Another suggestion of this general nature is [45.54] to insert an all pass filter that introduces a frequency-dependent group delay whose magnitude is randomly varying in time. Again, because of the small amount of delay introduced, this is equivalent to adding a frequency shaped noise.

A very effective way of dealing with the nonuniqueness was proposed in [45.55]. The proposal is to *distort* the signals $x_1(n)$ and $x_2(n)$ by passing each through a zero-memory nonlinearity. The rationale for this rather drastic proposal is as follows. Consider the RLS algorithm. The optimal filter is obtained as the solution of (45.29). For the present discussion the recursive update of the estimate is not of interest. So to keep the notation simple we will omit the time index n on all vectors and matrices. The important point is that, if there is a single active source in the transmitting room, and the transmitting room impulse responses are of length $\leq L$, then the correlation matrix $\boldsymbol{R}$ is readily shown to be singular. This is a direct consequence of the linear relation between the signals $x_1(n)$ and $x_2(n)$ given in (45.39). Since room responses are infinitely long, as noted at the end of the last section, $\boldsymbol{R}$ is in principle not singular but highly ill-conditioned.

For the two-channel case the matrix $\boldsymbol{R}$ has the structure

$$\boldsymbol{R} = \begin{pmatrix} \boldsymbol{R}_{11} & \boldsymbol{R}_{12} \\ \boldsymbol{R}_{21} & \boldsymbol{R}_{22} \end{pmatrix} , \tag{45.44}$$

where the submatrices $\boldsymbol{R}_{pq} = \sum_{m=-\infty}^{n} \lambda^{n-m} \boldsymbol{x}_p(m) \boldsymbol{x}_q^{\mathrm{T}}(m)$, for $p, q = 1, 2$. Asymptotically, for large L these submatrices become circulant matrices. Hence it follows that

$$\boldsymbol{R}_{pq} = \boldsymbol{F}^{-1} \boldsymbol{S}_{pq} \boldsymbol{F} , \quad p, q = 1, 2 , \tag{45.45}$$

where $\boldsymbol{F}$ is the discrete Fourier transform matrix, and $\boldsymbol{S}_{pq}$ is a diagonal matrix whose diagonal elements $S_{pq}(f)$, $f = 0, \ldots, L-1$, constitute the discrete Fourier transform of the first row of $\boldsymbol{R}_{pq}$. Thus S_{11} and S_{22} are the auto spectra and S_{12} and S_{21} the cross spectra of $x_1(n)$ and $x_2(n)$.

This fact can be used to show that the eigenvalues of $\boldsymbol{R}$ are lower bounded by $1 - |\gamma(f)|^2$, where $\gamma(f)$ is the coherence function at frequency f, defined as

$$\gamma(f) = \frac{S_{12}(f)}{\sqrt{S_{11}(f) S_{22}(f)}} . \tag{45.46}$$

If $|\gamma(f)|$ becomes equal to 1 at any of the discrete frequencies, the matrix becomes singular.

The aim of decorrelating the signals $x_1(n)$ and $x_2(n)$ is to reduce $|\gamma(f)|$ at all f. Additive uncorrelated noise, as well as the time-varying all-pass filters, decreases the coherence function. However, as pointed out earlier the amount of additive noise needed for proper convergence is unacceptably high. What about the possibility of reducing the coherence by some *fixed transformation* of the signals? A linear transformation will not do because the coherence function is invariant under linear transformations. Hence the proposal is to try some nonlinear transformation.

We have given the above derivation in some detail because it turns out that a very simple zero-memory nonlinearity works rather well. That nonlinear transformation [45.55] is obtained by adding to the signal a full-wave rectified version of itself, scaled by a constant α that controls the amount of nonlinearity. The nonlinearity can be written

$$\tilde{x}_i(n) = x_i(n) + \alpha|x_i(n)| \,, \quad i = 1, 2 \,. \tag{45.47}$$

To deal with certain pathological situations (almost never encountered in practice) α is replaced by $-\alpha$ in one of the channels.

Such a nonlinearity allows convergence of the misalignment. Yet, somewhat surprisingly, for speech signals it is hardly perceptible for α even as large as 0.2. The reason we surmise is that voiced portions of a speech signal are rich in harmonics of the fundamental frequency; hence the distortion products are hidden under existing harmonics. Several other transformations have been tried [45.56] but have not shown any improvement over the simple nonlinearity shown in (45.47).

Although this nonlinear transformation provides a good solution to the nonuniqueness problem for teleconferencing, it does not provide a completely satisfactory solution for music signals. For complex musical signals consisting of several instruments playing together, the distortion is almost imperceptible, as in the case of speech signals. However, for certain music signals the nonlinearity introduces an unacceptable degradation. The signal from a flute, for instance, is almost a pure sinusoid. The distortion products for such a signal are quite perceptible.

Even after the introduction of the nonlinearity of (45.47) the reduction in coherence is fairly small in several regions of the spectrum. Hence for good performance a rapidly converging algorithm like RLS is still required.

An improved use of nonlinear distortion is presented in [45.56]. The distortion used is the one shown in (45.47), except that in one of the channels α is multiplied by a time-varying parameter σ. The magnitude of σ is unity, but its sign is flipped at every positive-going zero crossing of the signal. The signals $\alpha|x_1(n)|$ and $\sigma\alpha|x_2(n)|$ are shown to be uncorrelated with each other and with the signals $x_1(n)$ and $x_2(n)$. However, the main modification introduced is that *four* adaptive filters are used to model the echo instead of the usual two. Two of these filters, $\hat{h}_1$ and $\hat{h}_2$, have the signals $x_1(n)$ and $x_2(n)$ as inputs, and the other two filters, $\overline{h}_1$ and $\overline{h}_2$, have the signals $\alpha|x_1(n)|$ and $\sigma\alpha|x_2(n)|$, respectively, as inputs. The sum of the outputs of all four adaptive filters is subtracted from the echo to give the error signal that controls all four adaptive filters. It has been shown that the filters $\overline{h}_1$ and $\overline{h}_2$ adapt quite rapidly because their inputs are uncorrelated with each other as well as with $x_1(n)$ and $x_2(n)$. The actual cancelation is done by two additional filters whose coefficients are copied from the filters $\overline{h}_1$ and $\overline{h}_2$ in a manner similar to that used in the two-echo-path model described in Sect. 45.2.5. The adaptation is claimed to be rapid enough so that the NLMS algorithm is adequate. This advantage is offset by the fact that one now needs two more adaptive filters and two canceling filters.

45.4 Summary

This chapter dealt with the problem of echoes that arise during a telephone conversation. The traditional method for dealing with echoes on telephone circuits was a device known as an echo suppressor. It was shown that when echoes arrive with a long delay, e.g., in satellite communication, echo suppressors do not provide a satisfactory solution. During the 1960s a new device, called an echo *canceler*, was developed to deal with this problem. This chapter described the operation of such a device in great detail. Also discussed were more-recent applications of echo cancelers to single- and multichannel teleconferencing.

References

45.1 M.M. Sondhi: An adaptive echo canceler, Bell Syst. Tech. J. **46**, 497–511 (1967)

45.2 M.M. Sondhi: Echo Canceller, U.S. Patent 3499999 (1970), (filed Oct. 31, 1966).

45.3 J.L. Kelly, B.F. Logan: Self-adjust echo suppressor, U.S. Patent 3500000 (1970), (filed Oct. 31, 1966).

45.4 D.L. Duttweiler, Y.S. Chen: A single chip VLSI echo canceller, Bell Syst. Tech. J. **59**, 149–160 (1980)

45.5 R.R. Reisz, E.T. Klemmer: Subjective evaluation of delay and echo suppressors in telephone communications, Bell Syst. Tech. J. **42**, 2919–2943 (1963)

45.6 G. Williams, L.S. Moye: Subjective evaluation of unsuppressed echo in simulated long-delay telephone communications, Proc. Inst. Electr. Elect. **118**, 401–408 (1971)

45.7 B. Widrow, M.E. Hoff Jr.: Adaptive switching circuits, IRE Wescon Conf. Rec. (Institute of Radio Engineers, New York 1960) pp. 96–104, Part 4

45.8 S.P. Applebaum: Adaptive arrays, IEEE Trans. Antennas Propagation **24**, 573–598 (1976)

45.9 M.M. Sondhi, D. Mitra: New results on the performance of a well-known class of adaptive filters, Proc. IEEE **64**, 1583–1597 (1976)

45.10 S. Haykin: *Adaptive Filter Theory* (Prentice-Hall, Englewood Cliffs 1996) pp. 748–751

45.11 A. Weiss, D. Mitra: Digital adaptive filters: Conditions for convergence, rates of convergence, and effects of noise and errors arising from the implementation, IEEE Trans. Inform. Theory **25**, 637–652 (1979)

45.12 D.L. Duttweiler: A twelve-channel digital echo canceler, IEEE Trans. Commun. **26**, 647–653 (1978)

45.13 D.L. Duttweiler: Proportionate normalized least-mean-squares adaptation in echo cancelers, IEEE Trans. Speech Audio Process. **8**, 508–518 (2000)

45.14 J. Benesty, Y. Huang, J. Chen, P.A. Naylor: Adaptive algorithms for the identification of sparse impulse responses. In: *Topics in Acoustic Echo and Noise Control*, ed. by E. Haensler, G. Schmidt (Springer, Berlin, Heidelberg 2006)

45.15 J. Benesty, S.L. Gay: An improved PNLMS algorithm, Proc. IEEE ICASSP **2002**, 1881–1884 (2002)

45.16 S. Amari: Natural gradient works efficiently in learning, Neural Comput. **10**, 251–276 (1998)

45.17 S.L. Gay, S.C. Douglas: Normalized natural gradient adaptive filtering for sparse and nonsparse systems, Proc. IEEE ICASSP **2002**, 1405–1408 (2002)

45.18 T. Gaensler, S.L. Gay, M.M. Sondhi, J. Benesty: Double-talk robust fast converging algorithms for network echo cancellation, IEEE Trans. Speech Audio Process. **8**, 656–663 (2000)

45.19 R.W. Harris, D.M. Chabries, F.A. Bishop: A variable step (VS) adaptive filter algorithm, IEEE Trans. Acoust. Speech Signal Process. **34**, 309–316 (1986)

45.20 S. Marple: Efficient least-squares FIR system identification, IEEE Trans. Acoust. Speech Signal Process. **29**, 62–73 (1981)

45.21 L. Ljung, M. Morf, D.D. Falconer: Fast calculation of gain matrices for recursive estimation schemes, Int. J. Control **27**, 1–19 (1978)

45.22 D.D. Falconer, L. Ljung: Application of fast Kalman estimation to adaptive equalization, IEEE Trans. Commun. **26**, 1439–1446 (1978)

45.23 J. Cioffi, T. Kailath: Fast, recursive-least-squares transversal filters for adaptive filtering, IEEE Trans. Acoust. Speech Signal Process. **34**, 304–337 (1984)

45.24 E. Eweda: Comparison of RLS, LMS, and sign algorithms for tracking randomly time-varying channels, IEEE Trans. Signal Process. **42**, 2937–2944 (1994)

45.25 K. Ozeki, T. Umeda: An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties, Electron. Commun. Jpn. **67-A**, 19–27 (1984)

45.26 S.L. Gay: The fast affine projection algorithm. In: *Acoustic Signal Processing for Telecommunication*, ed. by S.L. Gay, J. Benesty (Kluwer Academic, Boston 2000), Chap. 2

45.27 S.G. Kratzer, D.R. Morgan: The partial-rank algorithm for adaptive beamforming, Proc. SPIE Int. Soc. Optic. Eng. **564**, 9–14 (1985)

45.28 S.L. Gay, S. Tavathia: The fast affine projection algorithm, Proc. IEEE ICASSP **1995**, 3023–3026 (1995)

45.29 E. Haensler: The hands-free telephone problem – an annotated bibliography, Signal Process. **27**, 259–271 (1992)

45.30 M.M. Sondhi, W. Kellermann: Adaptive echo cancellation for speech. In: *Advances in Speech Signal Processing*, ed. by S. Furui, M.M. Sondhi (Marcel Dekker, New York 1992), Chap. 11

45.31 I. Furukawa: A design of canceller of broad band acoustic echo, Int. Teleconf. Symp. (IEEE, New York 1984) pp. 232–239

45.32 W. Kellermann: Kompensation akustischer Echos in Frequenzteil-Bändern. In: *Proc. Aachener Kolloquium*, ed. by L. Butzer (RWTH Aachen, Aachen 1984) pp. 322–325, (in German)

45.33 W. Kellermann: Analysis and design of multirate systems for cancellation of acoustical echoes, Proc. IEEE ICASSP **1998**, 2570–2573 (1998)

45.34 P.P. Vaidyanathan: *Multirate Systems and Filter Banks* (Prentice-Hall, Englewood Cliffs 1992)

45.35 M. Vetterli: A theory of multirate filter banks, IEEE Trans. Acoust. Speech Signal Process. **35**, 356–372 (1987)

45.36 D.R. Morgan: Slow asymptotic convergence of LMS acoustic echo cancelers, IEEE Trans. Speech Audio Process. **3**, 126–136 (1995)

45.37 P.L. DeLeon, D.M. Etter: Experimental results with increased bandwidth analysis filters in over sampled sub-band acoustic echo cancelers, IEEE Signal Process. Lett. **2**, 1–3 (1995)

45.38 B. Hatty: Recursive least squares algorithms using multirate systems for cancellation of acoustical echoes, Proc. IEEE ICASSP **90**, 1145–1148 (1990)

45.39 D.R. Morgan, J. Thi: A delayless sub-band adaptive filter, IEEE Trans. Signal Process. **43**, 1819–1830 (1995)

45.40 J. Chen, H. Bes, J. Vandewalle, P. Janssens: A new structure for sub-band acoustic echo canceler, Proc. IEEE ICASSP **1988**, 2574–2577 (1988)

45.41 M. Dentino, J. McCool, B. Widrow: Adaptive filtering in the frequency domain, Proc. IEEE **66**, 1658–1659 (1978)

45.42 E.R. Ferrara Jr.: Fast implementation of LMS adaptive filter, IEEE Trans. Acoust. Speech Signal Process. **28**, 474–475 (1980)

45.43 D. Mansour, A.H. Gray Jr.: Unconstrained frequency-domain adaptive filter, IEEE Trans. Acoust. Speech Signal Process. **30**, 726–734 (1982)

45.44 J.-S. Soo, K.K. Pang: Multidelay block frequency domain adaptive filter, IEEE Trans. Acoust. Speech Signal Process. **38**, 373–376 (1990)

45.45 E. Moulines, O. Ait Amrane, Y. Grenier: The generalized multidelay adaptive filter: structure and convergence analysis, IEEE Trans. Signal Process. **43**, 14–28 (1995)

45.46 J. Benesty, T. Gaensler, D.R. Morgan, M.M. Sondhi, S.L. Gay: *Advances in Network and Acoustic Echo Cancellation* (Springer, Berlin, Heidelberg 2001)

45.47 K. Ochiai, T. Araseki, T. Ogihara: Echo canceler with two echo path models, IEEE Trans. Commun. **COM-25**(6), 589–595 (1977)

45.48 E.J. Diethorn: Improved decision logic for two-path echo cancelers. In: *Proc. IEEE Workshop on Acoustic Echo and Noise Control*, ed. by E. Haensler, P. Dreiseitel (IEEE, New York 2001)

45.49 S. Makino, Y. Kaneda, N. Koizumi: Exponentially weighted step size NLMS adaptive filter based on the statistics of a room impulse response, IEEE Trans. Speech Audio Process. **1**, 101–108 (1993)

45.50 E.J. Thomas: Some considerations on the application of Volterra representation of nonlinear networks to adaptive echo cancellers, Bell Syst. Tech. J. **50**, 2797–2805 (1971)

45.51 A. Stenger, L. Trautmann, R. Rabenstein: Nonlinear acoustic echo cancellation with 2nd order adaptive Volterra filters, Proc. IEEE ICASSP **1999**, 877–880 (1999)

45.52 M.M. Sondhi, D.R. Morgan, J.L. Hall: Stereophonic acoustic echo cancellation – an overview of the fundamental problem, IEEE Signal Process. Lett. **2**, 148–151 (1995)

45.53 Y. Joncour, A. Sugiyama: A stereo echo canceller with pre-processing for correct echo path identification, Proc. IEEE ICASSP **1998**, 3677–3680 (1998)

45.54 M. Ali: Stereophonic echo cancellation system using time-varying all-pass filtering for signal decorrelation, Proc. IEEE ICASSP **1998**, 3689–3692 (1998)

45.55 J. Benesty, D.R. Morgan, M.M. Sondhi: A better understanding and an improved solution to the specific problems of stereophonic acoustic echo cancellation, IEEE Trans. Speech Audio Process. **6**, 156–165 (1998)

45.56 S. Shimauchi, Y. Haneda, S. Makino, Y. Kaneda: New configuration for a stereo echo canceller with nonlinear pre-processing, Proc. IEEE ICASSP **1998**, 3685–3688 (1998)

46. Dereverberation

Y. Huang, J. Benesty, J. Chen

Room reverberation is one of the two major causes (the other is background noise) of speech degradation and there has been an increasing need for speech dereverberation in various speech processing and communication applications. Although it has been studied for decades, speech dereverberation remains a challenging problem from both a theoretical and practical perspective. This chapter provides a methodical overview of speech dereverberation algorithms. They are classified into three categories: source model-based speech enhancement algorithms for dereverberation, separation of speech and reverberation via homomorphic transformation, and speech dereverberation by channel inversion and equalization. We outline the basic ideas behind these approaches, explain the assumptions they make, and discuss their characteristics as well as performance.

46.1 Background and Overview

46.1.1 Why Speech Dereverberation?

Speech is probably the most natural and efficient form of human–human communication and arguably for human–machine interaction. But voice in rooms is subject to degradation caused by acoustic reverberation in addition to ambient noise.

Reverberation is the collection of reflected sounds from the surfaces in an enclosure. It adds *warmth* to sound, which is essential for music, and helps people better orient themselves in the listening environment. However, room reverberation leads to temporal and spectral smearing, which would distort both the envelope and fine structure of a speech signal. As a result, speech becomes difficult to be understood in the presence of room reverberation, especially for hearing-impaired and elderly people [46.1] and for automatic speech recognition systems [46.2, 3]. This gives rise to a strong need for effective speech dereverberation algorithms.

A speech dereverberation algorithm employing either one or multiple microphones is intended to suppress or compensate the effect of room reverberation such that the original clean speech can be faithfully recovered or its intelligibility is improved. Since neither the source speech signal nor the acoustic channel impulse responses are known a priori, speech dereverberation is a difficult, blind problem. This practical importance and theoretical challenge have inspired great amounts of research into this area in the last four decades. While significant progresses have been made, it is still far away from

claiming victory in the battle against room reverberation. This chapter provides an overview of the state of the art of research into speech dereverberation problem.

46.1.2 Room Acoustics and Reverberation Evaluation

Before we present various speech dereverberation methods, it is important to get an idea as to how reverberation impairs speech quality and how we can quantitatively describe the level of reverberation that proportionally indicates the amount of reduction in speech intelligibility.

Many psychoacoustic studies have reported that human perceptions of reflected sounds with short and long delays are different. It was clearly explained in [46.4] that, for long delays, a clutter of separate echoes is heard; but if the delay is short, the ear cannot distinguish the time difference between the direct and reverberant sounds. As such, a room impulse response can be broken into three parts: direct-path response, early and late reverberations, as illustrated in Fig. 46.1 with a sample acoustic impulse response measured in the varechoic chamber at Bell Labs. The cutoff time delay k_c with respect to direct sound between early and late reverberations is about 50 ms for speech and 80 ms for music.

Due to the so-called *precedence effect* [46.5], early reverberations tend to perceptually reinforce the direct sound and are therefore considered harmless to speech intelligibility. However, the frequency response of early reverberations is rarely flat such that the speech spectrum is distorted and a change in speech quality is perceived. This procedure is known as *coloration*.

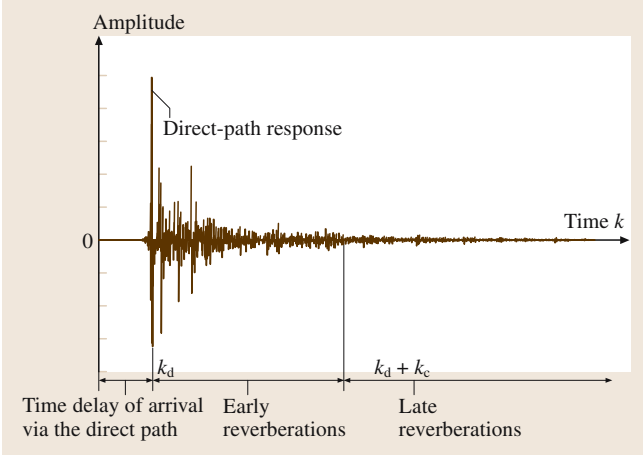


Fig. 46.1 Illustration of a sample acoustic impulse response measured in the varechoic chamber at Bell Labs

The impact of late reverberations on speech intelligibility can be interpreted from two perspectives. First, when examining the time sequence of a clean speech signal, one observes that the difference in level between a peak and its adjoining valley is evident. But after late reverberations add energy in the valleys between peaks, the difference becomes blurred and speech intelligibility gets worse. Second, reverberant sounds with long delays can stretch out to the following syllables or even next word and mask them as interference, which causes difficulty in understanding the meaning that those syllables and words convey. The degree of difficulty varies, depends on how long reverberant sounds persist and how strong they are relatively to the direct sound.

It is then easy to understand why the level of reverberation is typically measured by the reverberation time, which was introduced by *Sabine* in [46.6] and is defined as the length of time that it takes the reverberation to decay 60 dB from the level of the original sound. For measuring (or more precisely, estimating) the reverberation time of an acoustic impulse response, *Schroeder*'s backward integration [46.7] is probably the most widely used method; a general guideline was suggested in [46.8] as to how the measured impulse response can be accurately truncated at the knee where the main decay slope intersects the noise floor. The reverberation time for a typical room in a home or office ranges from 100 ms (slightly reverberant) to 600 ms (highly reverberant).

Another reverberation measure as an objective predicator of speech intelligibility is the early to late energy ratio (ELER) [46.9], which is defined in dB as follows:

$$\mathrm{ELER} \triangleq 10\log_{10}\left[\frac{\sum_{k=k_d}^{k_d+k_c-1} h^2(k)}{\sum_{k=k_d+k_c}^{k_d+L_h-1} h^2(k)}\right], \qquad (46.1)$$

where $h(k)$ is a discrete-time acoustic impulse response, k_d is the time delay of arrival via the direct path, k_c corresponds to the cutoff time delay separating early and late reverberations, and L_h is the length of the impulse response. This ratio is also known as the *clarity index*.

46.1.3 Classification of Speech Dereverberation Methods

There are different ways to classify various existing speech dereverberation algorithms, depending on what one is interested in. With the number of microphones as the feature to focus on, speech dereverberation can be classified into the classes of single- and multichan-

nel methods. But in this chapter, we care more about the underlying principles and therefore choose to classify speech dereverberation methods into the following three groups: source model-based speech dereverberation, separation of speech and reverberation via homomorphic transformation, and speech dereverberation by channel inversion and equalization. This classification is reflected by the structure of this chapter.

Source model-based speech dereverberation methods compute an estimate of the clean speech by using our a priori knowledge about the structure of clean speech signals and how they are distorted by room reverberation. Typical algorithms in this group include linear-predication (LP) residual enhancement methods [46.10–13], harmonic filtering [46.14], and speech dereverberation using probabilistic models [46.15]. They employ either speech production model or other statistical speech models. In Sect. 46.3, we will discuss these algorithms.

The reverberation process in the time domain is mathematically depicted as a convolution of a clean speech sequence with an unknown room impulse response. In the second group of methods, speech dereverberation is carried out from the perspective of signal separation in the light of cepstral analysis, which is a homomorphic transformation (see [46.16, Chap. 12] and the references therein for a more complete description of homomorphic systems and cepstral analysis). Since signals that are convolved in the time domain have complex cepstra that are additively combined, speech and reverberation can be more feasibly separated in the cepstral domain. There are two major techniques in this group: cepstral liftering [46.17–20], and cepstral mean subtraction/high-pass filtering of cepstral frame coefficients [46.21–23], which will be studied in Sect. 46.4.

Speech dereverberation by channel inversion and equalization seeks to estimate an inverse filter of the acoustic impulse response. This is a commonly used deconvolution approach. But the main problem is that the acoustic impulse response must be known, or blindly estimated, for successful dereverberation. Since there is a significant difference in difficulty and complexity of channel estimation and inversion for single- and multichannel systems, we further divide this class of speech dereverberation methods into the techniques for single- and multichannel systems. For a single-channel acoustic system, we have direct inverse, minimum mean square error (MMSE), and least-squares algorithms. For a multichannel acoustic system, we can use either beamforming [46.24], matched filtering [46.25], or multichannel inverse theorem (MINT) method [46.26]. All these algorithms will be developed in Sect. 46.5.

46.2 Signal Model and Problem Formulation

In a reverberant environment, the signal that a single microphone captures is the convolution of the source speech signal and the corresponding channel impulse response plus additive noise:

$$x(k) = s(k) * h + b(k) \,, \tag{46.2}$$

where $x(k)$, $s(k)$, and $b(k)$ are the microphone output, source speech signal, and background noise at time k, respectively, h is the channel impulse response, and $*$ denotes linear convolution. In acoustic signal processing, the channel impulse response is usually modeled as an finite impulse response (FIR) filter [46.27]. Moreover, since room acoustics change slower than speech, it is usually assumed that the channel impulse response is time invariant (at least short-time time invariant) for speech dereverberation. As a result, for ease of presentation and discussion, h in (46.2) is not considered as a function of time. Putting (46.2) into vector/matrix form, we get

$$x(k) = \boldsymbol{h}^{\mathrm{T}} \boldsymbol{s}(k) + b(k) \,, \tag{46.3}$$

where

$$\boldsymbol{h} = [\, h(0) \;\; h(1) \;\; \cdots \;\; h(L_h - 1) \,]^{\mathrm{T}} \,,$$
$$\boldsymbol{s}(k) = [\, s(k) \;\; s(k-1) \;\; \cdots \;\; s(k - L_h + 1) \,]^{\mathrm{T}} \,,$$

and L_h is the length of h.

When multiple microphones are available, we have a single-input multiple-output (SIMO) system and, similar to (46.3), each microphone output is given by

$$x_n(k) = \boldsymbol{h}_n^{\mathrm{T}} \boldsymbol{s}(k) + b_n(k) \,, \quad n = 1, 2, \cdots, N \,, \tag{46.4}$$

where N is the number of microphones, $x_n(k)$, $\boldsymbol{h}_n$, and $b_n(k)$ with respect to microphone n are similarly defined as in the case of single microphone.

The task of speech dereverberation is to recover $s(k)$, given only the observed microphone signal $x(k)$ or signals $x_n(k)$ $(n = 1, 2, \cdots, N)$, without assuming any a priori knowledge regarding h or h_n, respectively.

46.3 Source Model-Based Speech Dereverberation

Signal modeling is based on our knowledge about the signal source and has been found very useful for various signal processing and communication systems. In particular, source models are important to signal enhancement. A good signal model provides a concise and insightful description of how the signal is produced. Consequently, when applying the model, we enhance the signal in a smaller space of the model parameters such that a practical and effective system can be implemented in a very efficient manner. Following this line of ideas, we will study source model-based speech dereverberation methods first in this section.

46.3.1 Speech Models

Speech Production Model

Speech is produced by our articulation systems, which can be abstracted by a source-vocal-tract model. The traditional framework for speech analysis is centered around the speech production model first proposed by Homer *Dudley* at Bell Laboratories in the 1930s [46.28]. In this model, as depicted in Fig. 46.2, speech is produced by passing an excitation through a time-varying all-pole filter. The coefficients of the all-pole filter hinge on the shape of the vocal tract for the specific sound being generated. The excitation signal $e(k)$ is either a pulse train for voiced speech or random noise for unvoiced sound. The speech signal is then written as

$$s(k) = \sum_{p=1}^{P} a_{\mathrm{p}} s(k-p) + e(k) \,, \tag{46.5}$$

where P and the a_{p} are the order and the coefficients of the all-pole filter, respectively. By analyzing $s(k)$ with linear prediction [46.29,30], we can determine a_{p} and the excitation $e(k)$, which is also called LP residual signal.

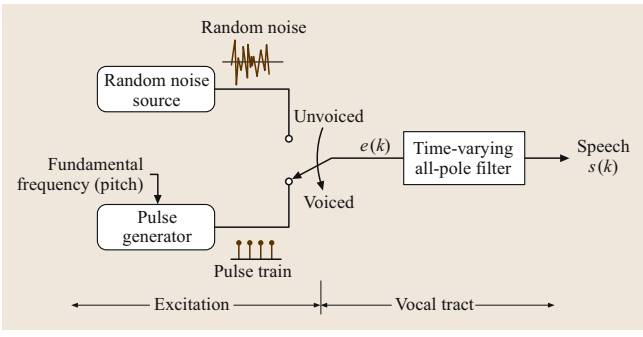


Fig. 46.2 The speech production model

Statistical Speech Models

Speech is an intrinsically random process. Therefore, in addition to describing a speech signal in a deterministic manner, we can alternatively choose to characterize speech with a statistical model. Nowadays, the most successful statistical speech model is the hidden Markov model (HMM). Its merits were recognized as the statistical pattern matching based on Bayes' theorem becomes a dominant approach to speech recognition. HMM keenly captures two levels of variation: one due to the uncertainty in the realization of a particular sound and the other due to the uncertainty in the sequential changes from one sound to another [46.31]. But HMM is too complex a model to be employed in a real-time speech dereverberation system.

For speech dereverberation, a simplified statistical speech model such as that proposed in [46.15] is desired. This is a probabilistic speech production model in which the joint probability density function (PDF) of a frame of clean speech samples $\boldsymbol{s}$, the LP feature vector $\boldsymbol{\theta} \triangleq [a_1, a_2, \cdots, a_P, \sigma_{\mathrm{e}}]^{\mathrm{T}}$, and the index c of a cluster of speech frame is expressed using the chain rule as follows:

$$p(\boldsymbol{s}, \boldsymbol{\theta}, c) = p(\boldsymbol{s}|\boldsymbol{\theta}, c)\, p(\boldsymbol{\theta}|c)\, p(c) \,. \tag{46.6}$$

Note that here we omit the frame index for ease of presentation. Moreover, the excitation is assumed to be a zero-mean Gaussian random process with variance σ_{e}^2 and then the conditional PDF $p(\boldsymbol{s}|\boldsymbol{\theta}, c)$ is calculated using the speech production model (46.5) as follows:

$$p(\boldsymbol{s}|\boldsymbol{\theta}, c) = p(e = x * a_{\mathrm{LPE}}) \sim \mathcal{N}\left(0, \sigma_{\mathrm{e}}^2\right) \,, \tag{46.7}$$

where a_{LPE} is the linear prediction error filter whose transfer function is given by

$$A_{\mathrm{LPE}}(z) = 1 - \sum_{p=1}^{P} a_{\mathrm{p}} z^{-p} \,,$$

and $\mathcal{N}(0, \sigma_{\mathrm{e}}^2)$ denotes a Gaussian distribution with mean 0 and variance σ_{e}^2. The LP coefficients and excitation are considered independent such that

$$p(\boldsymbol{\theta}|c) = p(a_1, a_2, \cdots, a_P|c) \cdot p(\sigma_{\mathrm{e}}|c) \,. \tag{46.8}$$

It is further asserted that the LP coefficients are Gaussian distributed and the nonnegative random variable σ_{e} is Gamma distributed. The parameters of the two distributions depend on the cluster. They are estimated by training and so is the a priori distribution $p(c)$. In [46.15], the cluster size was 256.

Finally, the distribution of a clean speech frame is obtained:

$$p(\boldsymbol{s}) = \sum_{c} \left[\int_{\boldsymbol{\theta}} p(\boldsymbol{s}, \boldsymbol{\theta}, c)\, \mathrm{d}\boldsymbol{\theta} \right] . \qquad (46.9)$$

For simplicity, the integration over all possible feature vectors ($\boldsymbol{\theta}$) and the summation over all speech clusters (c) are omitted in [46.15], which leads to

$$p(\boldsymbol{s}) \approx p(\boldsymbol{s}, \boldsymbol{\theta}, c) = p(\boldsymbol{s}|\boldsymbol{\theta}, c)\, p(\boldsymbol{\theta}|c)\, p(c) . \qquad (46.10)$$

46.3.2 LP Residual Enhancement Methods

According to the speech production model, speech is produced as an all-pole filter is excited by a pulse train or noise-like signal. Since an acoustic channel is usually modeled as an FIR filter whose transfer function consists of only zeros for speech dereverberation and many other acoustic signal processing problems, it was assumed that room reverberation would introduce only zeros into the microphone signals and, as a result, would primarily affect only the nature of the speech excitation sequence, having little impact on the all-pole filter [46.10]. Therefore, speech dereverberation can be accomplished by leaving the LP coefficients untouched and processing only the speech excitation signal obtained as the residual following LP analysis, as illustrated in Fig. 46.3.

LP residual of voiced speech is a well-structured pulse train, which would be smeared when room reverberation is present. Therefore reverberation effect can be clearly identified and readily eliminated in these periods of time. There have been a number of LP residual enhancement methods in the literature of speech dereverberation, which are based on this idea. In the following, we will review these methods in the order of their publication date:

Wavelet Transform Domain LP Residual Enhancement [46.10]

Brandstein and *Griebel* observed that a wavelet is the derivative of a smoothing function, which leads to the fact that a wavelet transform will contain local extrema at large changes in the examined signal. Therefore, they argued that impulses in the residual signal of voiced speech will manifest as extrema are detected. The desired wavelet extrema that correspond to the impulses in the clean LP residual are estimated via a clustering method and finally the LP residual sequence is enhanced.

Although the significance of the wavelet extrema in the unvoiced speech segments is not as clearly understood, the authors reported that this wavelet transform

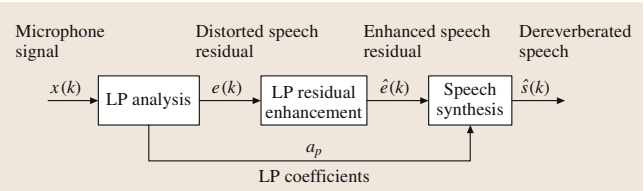


Fig. 46.3 Block diagram of an LP residual enhancement method for speech dereverberation

domain LP residual enhancement method is also effective in reconstructing a noise-like residual signal.

LP Residual Enhancement via Kurtosis Maximization [46.13]

Speech is a super-Gaussian random process whose kurtosis is greater than 3. [Kurtosis (Pearson kurtosis) is the fourth central moment divided by the fourth power of the standard deviation. Kurtosis measures the degree of peakedness of a distribution. The Gaussian distribution has a moderate tail and its kurtosis is 3. A super-Gaussian distribution has a high peak around its mean and a small tail, leading to a large kurtosis greater than 3. On the contrary, a sub-Gaussian distribution with a flat-topped curve and a long tail has a small kurtosis less than 3.] In the presence of room reverberation, a microphone signal is the mixture of the source speech signal and its delayed/attenuated copies. According to the central limit theorem, a mixture of independent non-Gaussian random variables is more like a Gaussian-distributed signal. Therefore, clean speech has a larger kurtosis compared to its reverberant observations. *Gillespie* et al. in [46.13] proposed to use kurtosis as a reverberation metric and developed an LP residual enhancement algorithm via kurtosis maximization.

LP Residual Enhancement via Hilbert Envelope Weighting [46.12]

Yegnanarayana et al. suggested to use the Hilbert envelope for LP residual reconstruction. For an LP residual signal $e(k)$, its Hilbert envelope is computed as

$$e_{\mathrm{HE}}(k) = \sqrt{e^2(k) + e_{\mathrm{H}}^2(k)} , \qquad (46.11)$$

where $e_{\mathrm{H}}(k)$ denotes the Hilbert transform of $e(k)$. The Hilbert transform is obtained by switching the real and imaginary parts of the discrete Fourier transform (DFT) of $e(k)$ and then taking the inverse DFT. The feature that the Hilbert envelope has a large amplitude at the instant of strong excitation makes it a good indicator of glottal closure, where an excitation pulse takes place. Therefore, applying the Hilbert envelope to an LP residual as

a weighting function has the effect of emphasizing the pulse train structure for voiced speech, which leads to an enhanced LP residual signal with less reverberation.

When there are multiple microphones, the Hilbert envelopes of the LP residuals from all microphone outputs can be *coherently* added as follows

$$e_{\mathrm{ca},\,1}(k) = \sqrt{\sum_{n=1}^{N} e_{\mathrm{HE}}^2(k+\tau_{1n})}\,, \tag{46.12}$$

where $e_{\mathrm{ca},\,1}(k)$ is the coherent average of the Hilbert envelopes with respect to microphone 1, and τ_{1n} is the relative time difference of arrival between microphone 1 and n. This average can further suppress large amplitude spikes in the Hilbert envelope caused by reverberation and therefore can improve the performance of speech dereverberation.

Discussion

Speech dereverberation by LP residual enhancement is motivated by the assumption that room reverberation would have very little impact on the estimated vocal-tract responses but distort only the residual signal of LP analysis. Although intuitively making some sense, the assumption has not yet been justified in practice and, on the contrary, it disagrees somewhat with our experience. As shown by a comparison of a clean speech signal and its reverberant microphone output in Fig. 46.4, their vocal-tract responses estimated by LP analysis are significantly different: even the first two formants have been altered by room reverberation most of the time.

However, the inaccuracy of the assumption does not lead to the imprudent conclusion that algorithms based on LP residual enhancement are futile or ineffective for speech dereverberation. From a signal processing perspective, LP analysis essentially prewhitens the microphone outputs. Its impact on the signal will be compensated by the inverse LP filter at the end. Therefore the key point to success for this class of methods is whether room reverberation could be much more easily offset by processing the prewhitened LP residual signals. The LP enhancement algorithms presented in this section can remove strong reverberations at best. They are quite helpless in combating weak reverberations in the tail of an acoustic impulse response, which unfortunately are also harmful to speech intelligibility. In addition, speech artifacts are usually introduced, which makes the dereverberated speech sound unnatural. Therefore, LP residual enhancement-based speech dereverberation methods can only achieve moderate successes in relatively low-reverberant environments.

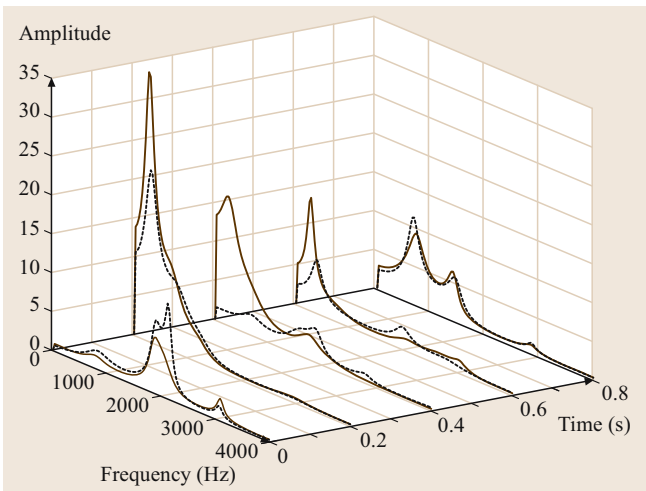


Fig. 46.4 Comparison of the all-pole filter responses for a clean speech signal (*solid line*) and its microphone output (*dotted line*) obtained by convolving the clean speech with an acoustic impulse response measured in a reverberant environment with an approximately 580 ms T_{60} reverberation time in the absence of noise. The sampling rate is 8 kHz and the two signals have been properly realigned prior to LP analysis. The order of the all-pole filter is $P = 10$. The frame size is 10 ms and the analysis window is 30 ms long with 2/3 overlapping. The results of only five frames are plotted here

46.3.3 Harmonic Filtering

Harmonic structure is an important attribute of speech, particularly voiced speech whose excitation is produced by the vibrating vocal cord. In the frequency domain, the spectrum of a voiced speech signal is composed of a fundamental frequency F_0 (i. e., pitch) and a series of harmonics. As long as the harmonics are precise multiples of the fundamental, the voice sounds clear and pleasant. But when nonharmonic components are added as room reverberation possibly does, varying degrees of roughness, harshness, or hoarseness will be perceived.

Motivated by this psychoacoustic fact, *Nakatani* et al. proposed in [46.14] to conduct speech dereverberation by suppressing the nonharmonic components and reconstructing the damaged harmonic structures. In such an algorithm, it is critical to estimate the fundamental frequency of a reverberant speech signal accurately, which itself is not an easy problem, as discussed in detail in [46.32]. After the F_0 is determined, the harmonic components are extracted from the speech spectrum to form an enhanced speech spectrum (for voiced speech segments only). By comparing the en-

hanced and corresponding reverberant speech spectrum, a dereverberation filter is calculated. A mean dereverberation filter is obtained by averaging the dereverberation filters of different voiced speech segments. Finally, applying the mean dereverberation filter to the entire reverberant speech sequence yields the dereverberated signal. It is suggested to repeat this procedure once more to obtain better performance.

Discussion

Harmonic filtering is a perception-based speech enhancement method for dereverberation. It eliminates the nonharmonic components, which contain both speech and reverberation even in the periods of voiced speech. Consequently, artifacts can be perceived in the enhanced signal. In addition, the dereverberation filter is estimated from voiced speech segments. Its performance for unvoiced speech segments cannot be theoretically analyzed and predicted. Moreover, the harmonic filter method ignores the reverberation effect on the harmonic components, which certainly prevents perfect dereverberation even in a presumably ideal case. Finally, this is an offline technique and cannot be implemented in real time. This makes it less attractive for many speech processing and communication applications.

46.3.4 Speech Dereverberation Using Probabilistic Models

LP residual enhancement and harmonic filtering methods are all based on the *deterministic* speech production model. In [46.15], *Attias* et al. attempted to use the *probabilistic* models described previously in Sect. 46.3.1 for speech dereverberation.

Given a frame of the microphone output $\boldsymbol{x}$, speech dereverberation is carried out by estimating a frame of speech signal $\hat{\boldsymbol{s}}$ that gives the largest posterior probability $p(\boldsymbol{s}|\boldsymbol{x})$:

$$\hat{\boldsymbol{s}} = \arg\max_{\boldsymbol{s}} p(\boldsymbol{s}|\boldsymbol{x}) \,. \tag{46.13}$$

Since

$$p(\boldsymbol{s}|\boldsymbol{x}) = \frac{p(\boldsymbol{x},\boldsymbol{s})}{p(\boldsymbol{x})} \,, \tag{46.14}$$

maximizing $p(\boldsymbol{s}|\boldsymbol{x})$ is equivalent to maximizing the joint distribution $p(\boldsymbol{x},\boldsymbol{s})$. By using the chain rule and incorporating the probabilistic clean speech model (46.10), we have

$$p(\boldsymbol{x},\boldsymbol{s}) = p(\boldsymbol{x}|\boldsymbol{s})p(\boldsymbol{s}) \approx p(\boldsymbol{x}|\boldsymbol{s})p(\boldsymbol{s}|\boldsymbol{\theta},c)p(\boldsymbol{\theta}|c)p(c) \,. \tag{46.15}$$

The a priori distribution $p(\boldsymbol{x}|\boldsymbol{s})$ is computed by using the signal model given by (46.2) and assuming the knowledge of the distribution of the additive noise:

$$p(\boldsymbol{x}|\boldsymbol{s}) = p(b = x - h * s) \,. \tag{46.16}$$

If the additive noise is Gaussian distributed with zero mean and variance σ_{b}^2, then

$$p(x - h * s) \sim \mathcal{N}\left(0, \sigma_{\mathrm{b}}^2\right) \,. \tag{46.17}$$

Clearly, the joint distribution $p(\boldsymbol{x},\boldsymbol{s})$ depends on the speech signal's characteristics $(\boldsymbol{s},\boldsymbol{\theta},c)$ and the acoustic channel conditions $(h,\sigma_{\mathrm{b}}^2)$. While $\boldsymbol{s}$ is the target, the other variables also need to be estimated. In [46.15], the iterative expectation-maximization (EM) algorithm was employed.

In the E-step, the algorithm computes the expected value of the clean speech using the current estimate of the acoustic channel conditions and the microphone signal

$$\hat{\boldsymbol{s}}(\kappa+1) = E\left\{\boldsymbol{s}\,\middle|\,\boldsymbol{x},\hat{h}(\kappa),\hat{\sigma}_{\mathrm{b}}^2(\kappa)\right\} \,, \tag{46.18}$$

where $E\{\cdot\}$ denotes mathematical expectation and κ is the iteration step index. In the M-step, the algorithm uses the data from the E-step, as if it were clean speech, to determine a maximum-likelihood estimate of the acoustic channel conditions:

$$\begin{aligned}&\left[\hat{h}(\kappa+1),\hat{\sigma}_{\mathrm{b}}^2(\kappa+1)\right]\\ &= \arg\max_{h,\sigma_{\mathrm{b}}^2} p\left[\hat{\boldsymbol{s}}(\kappa+1)|\boldsymbol{x},h,\sigma_{\mathrm{b}}^2\right] \,.\end{aligned} \tag{46.19}$$

The EM algorithm iterates (46.18) and (46.19) until convergence.

Discussion

While the algorithm presented in this subsection looks sophisticated, its essence is not difficult to be comprehended: as previously explained, speech dereverberation is challenging since it is a blind problem where neither the clean speech nor the channel impulse response is known; if an estimated speech distribution (which can be achieved by training) is provided, then we can process the microphone signals such that the enhanced signal matches the speech distribution well. The E-step performs essentially speech recognition in which the knowledge of the acoustic channel is assumed (the channel impulse response estimated in the previous step is used). Using the recognized speech, the algorithm then tries to estimate the acoustic channel impulse response in the M-step, which is a typical system identification problem where a reference signal is available. After

convergence, not only has speech been dereverberated, but also an estimate of the acoustic channel impulse response can be generated.

Although the idea appears technically sound, its effectiveness for speech dereverberation has not yet been justified with simulations. Speech recognition improvement in terms of accuracy for a common recognizer processing speech signals with and without dereverberation (one of the key performance measures) was unfortunately not reported. Statistical model-based speech dereverberation relies largely on the accuracy of the model, but one needs to trade off model accuracy for intractability. For example, the number of speech clusters has to be moderate. It is unexpected that such an algorithm can work well without careful tuning, which is similar to speech recognition systems.

The statistical speech model-based dereverberation method produces an estimate of the acoustic channel impulse response as a byproduct. From the perspective of blind system identification, this algorithm actually uses higher-order statistics (HOS) to estimate the channel impulse responses. But computing reliable HOS needs a great amount of data samples, which together with intensive computational complexity, makes the algorithm difficult to implement in real-time systems.

46.4 Separation of Speech and Reverberation via Homomorphic Transformation

Homomorphic transformation is a well-known signal processing method, which transforms nonadditively combined signals into a vector space in which the transformed signals are additively mixed [46.16]. To process signals that have been combined by convolution, cepstrum analysis is the most widely used homomorphic transformation. For a signal $x(k)$, its complex cepstrum (subscript 'c') is defined as

$$x_c(k) = \mathrm{IFFT}\{\ln[\mathrm{FFT}\{x(k)\}]\} , \tag{46.20}$$

where $\mathrm{FFT}\{\cdot\}$ and $\mathrm{IFFT}\{\cdot\}$ are the fast Fourier and inverse fast Fourier transforms, respectively. If $x(k) = h * s(k)$, then in the cepstral domain we have

$$x_c(k) = h_c + s_c(k) , \tag{46.21}$$

and we can separate $s_c(k)$ and h_c by means of linear filtering. The cepstrum transform was found to be very useful for deconvolution and has also been applied to speech dereverberation, particularly in robust speech recognition systems.

46.4.1 Cepstral Liftering

In order to separate the complex cepstra of a clean speech signal and an acoustic channel impulse response, we need to look for a property by which the two components can be differentiated. It is believed from observations that speech's cepstrum only has *low-time* components concentrated around the cepstral origin, whereas a typical acoustic channel impulse response is characterized by pulses with rapid ripples located far away from the cepstral origin. Therefore, an estimate of the speech cepstrum $\hat{s}_c(k)$ can be obtained by *low-time* liftering the microphone signal's cepstrum. (Low-time liftering refers to the operation that keeps low-time while suppressing high-time cepstral components. Low-time liftering in the cepstral domain is analogous to low-pass filtering in the frequency domain.)

Cepstral liftering relies on empirical evidence and uses a number of heuristics to determine the proper cutoff time for low-time liftering [46.17, 18]. Common problems with this kind of speech dereverberation techniques include inconsistent performance and the introduction of new distortion to the speech signal. Therefore, it has little use in real-world speech dereverberation systems for speech communication. The ideal of cepstral liftering (or log-spectral filtering) was also proposed for speech dereverberation in early speech recognition systems, include bandpass liftering [46.19] and cepstral weighting [46.20]. However, their effectiveness is also limited.

46.4.2 Cepstral Mean Subtraction and High-Pass Filtering of Cepstral Frame Coefficients

While cepstral liftering is an intraframe technique for speech dereverberation, interframe features of speech and acoustic room impulse responses can also be exploited to separate speech from reverberation. Since speech usually varies at a faster rate than acoustic room impulse responses, the channel impulse response is reasonably considered as stationary over short times. Then

taking the average of (46.21) over a proper duration of time yields

$$\bar{x}_c = h_c + \bar{s}_c \,, \tag{46.22}$$

where $\bar{x}_c$ and $\bar{s}_c$ are the short-time means of $x_c(k)$ and $s_c(k)$, respectively. In cepstral mean subtraction (CMS) [46.21], the short-time mean of speech cepstrum is assumed to be zero, which leads to $\bar{x}_c = h_c$. Therefore the dereverberated speech cepstrum will be given as

$$\hat{s}_c(k) = x_c(k) - \bar{x}_c \,. \tag{46.23}$$

Subtracting the short-time cepstral mean removes the direct-current (DC) component while keeping their quickly varying counterparts. This idea can easily be generalized to high-pass filtering of cepstral frame coefficients, which is known as the relative spectral processing (RASTA) method [46.22].

Both the CMS and RASTA methods assume that the speech cepstrum has zero mean, which is not precisely so in practice. Observing this drawback, Rahim and Juang proposed the signal bias removal (SBR) algorithm in which speech cepstrum vectors are classified into a number of clusters and each cluster has a different speech cepstral mean estimate [46.23]. Compared to the CMS and RASTA, the SBR algorithm is more computationally intensive and is hence less popular in robust speech recognition systems combating room reverberation and channel distortion.

All these cepstrum domain algorithms were proposed from the speech recognition research community. While various levels of speech dereverberation effect were reported for speech recognizers, they are not very effective for speech communication where the human auditory system is the target.

46.5 Channel Inversion and Equalization

Although it is debatable whether it is necessary in practice, perfect speech dereverberation is definitely attractive and can be achieved probably only by traditional channel inversion and equalization schemes. In this section, we will review existing channel inversion algorithms for speech dereverberation. Techniques for single- and multichannel systems will be separately developed. We focus merely on linear channel equalizers, particularly their stability, causality, and robustness.

46.5.1 Single-Channel Systems

Direct Inverse

Among all existing channel inversion methods, the most straightforward is the direct inverse method. This method assumes that the acoustic channel impulse response is known or has already been estimated. Then, as shown in Fig. 46.5a, the equalizer filter is determined by inverting the channel transfer function $H(z)$, which is the z-transform of the channel impulse response:

$$G(z) = \frac{1}{H(z)} \,. \tag{46.24}$$

In practice, the inverse filter g needs to be stable and causal. It is well known that the poles of a stable, causal, and rational system must be inside the unit circle in the z-plane. As a result, a stable, causal system has a stable and causal inverse only if both its poles and zeros are inside the unit circle. Such a system is commonly referred to as a *minimum-phase* system [46.16]. Although many systems are minimum phase, room acoustic impulse responses are unfortunately almost never minimum phase [46.33]. Consequently, while a stable inverse filter can still be found by using an all-pass filter, the inverse filter will be infinite impulse response (IIR), which is noncausal and has a long delay. In addition, inverting a transfer function is sensitive to estimation errors in the channel impulse response, particularly at those frequencies where the channel transfer function has a small

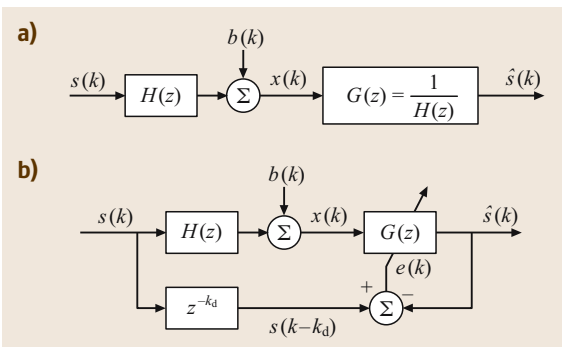


Fig. 46.5a,b Illustration of two well-known channel equalization approaches to speech dereverberation for a single-channel system: **(a)** direct inverse (or zero forcing) and **(b)** minimum mean square error (or least-squares) methods

amplitude. These drawbacks make direct-inverse equalizers impracticable for real-time speech dereverberation systems.

Minimum Mean-Square Error (MMSE) and Least-Squares (LS) Methods

If a reference source signal rather than an estimate of the acoustic channel impulse response is available, we can directly apply a linear equalizer to the microphone signal and adjust the equalizer coefficients such that the output can be as close to the reference as possible, as shown in Fig. 46.5b. The error signal is defined as

$$e(k) \triangleq s(k-d) - \hat{s}(k) = s(k-d) - g * x(k) \,, \quad (46.25)$$

where d is the decision delay for the equalizer. Then the equalization filter g is determined as the one that either minimizes the mean square error or yields the least squares of the error signal:

$$\hat{g}_{\mathrm{MMSE}} = \arg\min_{g} E\{e^2(k)\} \,, \quad (46.26)$$

$$\hat{g}_{\mathrm{LS}} = \arg\min_{g} \sum_{k=0}^{K-1} e^2(k) \,, \quad (46.27)$$

where K is the number of observed data samples. This is a typical problem in estimation theory. The solution can be found with well-known adaptive or recursive algorithms.

For minimum-phase single-channel systems, it can be shown that the MMSE/LS equalizer is the same as the direct-inverse or zero-forcing equalizer. But for non-minimum-phase acoustic systems, the MMSE/LS method essentially equalizes the channel by inverting only those components whose zeros are inside the unit circle [46.26]. In addition, it is clear that, for the MMSE/LS equalizer, a reference signal needs to be accessible. However, although the MMSE/LS method has these limitations, it is quite useful in practice and has been successfully applied for many speech dereverberation systems.

46.5.2 Multichannel Systems

While not much can be done about channel inversion when only one microphone is available, the use of multiple microphones opens up a wealth of new opportunities with speech dereverberation via channel inversion and equalization.

Beamforming

Beamforming is one of the oldest microphone array technique for speech acquisition and enhancement.

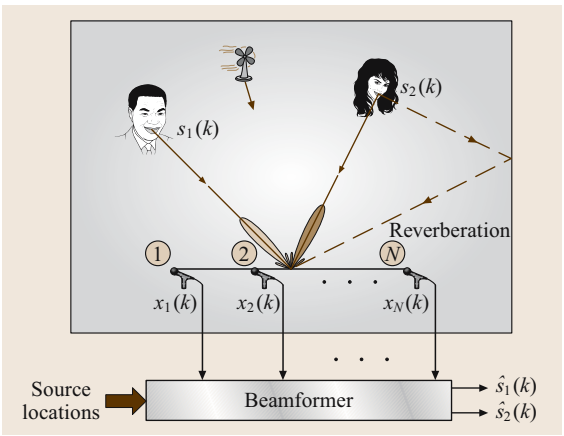


Fig. 46.6 Illustration of spatial filtering with a microphone array beamformer for speech acquisition and enhancement in an enclosed environment

A beamformer is a spatial filter that operates on the outputs of an array of microphones in order to enhance the signal coming from one direction while suppressing acoustic waves propagating from other directions, as illustrated in Fig. 46.6. While microphone array beamforming was primarily motivated to separate sound sources at different locations, it can effectively reduce the effect of room reverberation in the acquired speech signal.

Beamforming technique has two major types: delay-and-sum and filter-and-sum beamformers, as shown in Fig. 46.7. For a delay-and-sum beamformer, the estimate of a targeted speech signal is then given by

$$\hat{s}(k) = \sum_{n=1}^{N} g_n \cdot x_n(k - \tau_n) \,, \quad (46.28)$$

where the g_n are scalars and the delays τ_n are chosen such that the signals coming from the targeted source are aligned in phase in the output. A delay-and-sum beamformer is simple, but not very powerful. On one hand, it does not put nulls in the directions of the noise and interference sources whose positions are already known. On the other hand, the response of a delay-and-sum beamformer to an incoming signal depends not only on the signal's incident angle but also on the signal's spectrum. A delay-and-sum beamformer is essentially a narrowband beamformer, which does not have a uniform beam pattern over a wide frequency span. Consequently, if we use a delay-and-sum beamformer to capture speech that is a typical broadband signal, reverberations coming from a direction different from the beamformer's

look direction will not be equally attenuated over its entire spectrum. This *spectral tilt* results in a disturbing artifact in the array output.

In a filter-and-sum beamformer, as the name suggests, each microphone signal is first processed by a linear filter and then they are all summed to produce a dereverberated speech signal

$$\hat{s}(k) = \sum_{n=1}^{N} g_n * x_n(k) \,, \tag{46.29}$$

where g_n is the filter impulse response of length L_g for the n-th channel. Filter-and-sum beamformers are much more flexible and work well with wideband speech signals. The determination of the beamformer filters can be either data independent or data driven. In the former case, the filter coefficients and the beam pattern are relatively fixed. The look direction is steered to the direction of a specified target. In the latter case, the power of the beamformer output is minimized under the constraint that the signals coming from the look direction are aligned in phase or are distortionless, depending on how much is known about the acoustic channel from the target to the array of microphones. The linearly constrained minimum-variance (LCMV) algorithm developed by *Frost* [46.34] is the most studied and most influential algorithm of this kind. The generalized sidelobe canceler (GSC), developed by *Griffiths* and *Jim* [46.35], is rather an interesting way to implement the Frost's LCMV algorithm. The GSC algorithm transforms the LCMV beamforming from a constrained optimization problem into an unconstrained form. While they are essentially the same, the GSC has some implementation advantages, particularly for adaptive operation in time-varying acoustic environments (see [46.36] for a more-detailed discussion on the implementation advantages of the GSC algorithm). One main drawback of the GSC beamformer is that the cancelation of the target signals could occur in reverberant acoustic environments. Many robust variants of the GSC have been proposed to combat this problem [46.37–40].

Beamforming is a promising technique for speech dereverberation, but its performance is limited by a couple of factors in practice. Other than the issue of wideband speech signals, the problem of how to deal with near-field sound sources in reverberant acoustic environments still lacks of satisfactory solutions and even a decent framework for analysis. (Beamforming algorithms make a common assumption that the targeted sound source is in the far field of the array such that the wavefronts impinging on the array can be regarded as plane waves.)

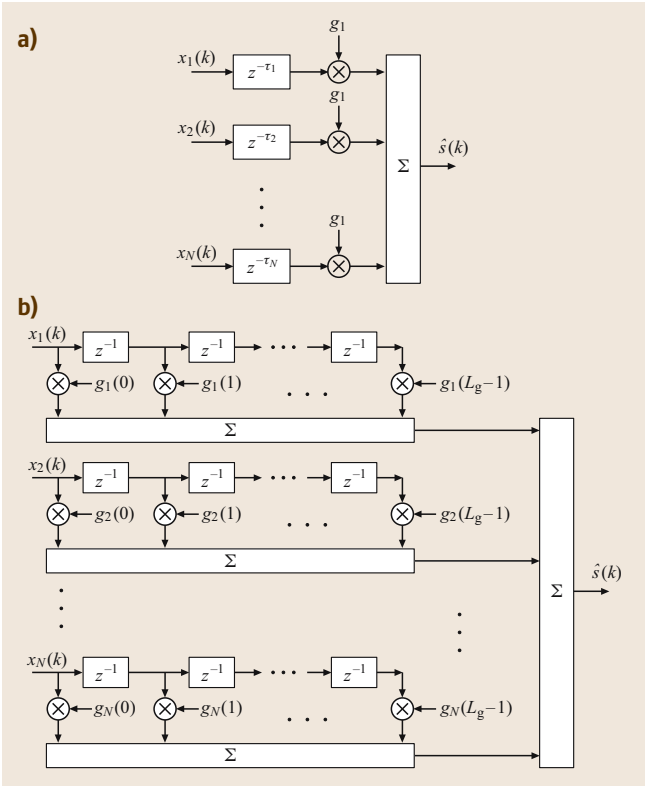


Fig. 46.7a,b The structure of **(a)** a delay-and-sum beamformer and **(b)** a filter-and-sum beamformer

Matched Filtering

In [46.25], *Flanagan* et al. proposed a matched filtering algorithm for speech acquisition and dereverberation as an alternative to beamforming. As opposed to knowing only the position of the speech source with a beamformer, the matched filtering algorithm assumes the availability of channel impulse responses from the targeted speech source to all microphones. Rather than finding inverse filters for those impulse responses, the algorithm convolved each microphone output with the time-reversed source to microphone impulse response, i. e., the matched filter. Then, without taking additive noise into account, the dereverberated speech signal is found as

$$\hat{s}(k) = \sum_{n=1}^{N} x_n(k) * h_n^{\mathrm{MF}} = s(k) * \left(\sum_{n=1}^{N} h_n * h_n^{\mathrm{MF}} \right) \,, \tag{46.30}$$

where

$$h_n^{\mathrm{MF}}(l) = \begin{cases} h_n(-l)\,, & -L_h < l \le 0 \\ 0\,, & \text{otherwise} \end{cases} \tag{46.31}$$

is the matched filter for the nth channel. It was shown in [46.25] that the matched-filtering algorithm is theoretically superior to the delay-and-sum beamformer. The gain in signal-to-reverberation ratio no longer depends on the number of reflections in the impulse response, but using full-length matched filters has a serious problem in practice. The convolution of h_n with h_n^{MF} is equivalent to the autocorrelation function of h_n, which has long tails on both the positive and negative sides. As a result, pre-echo (an attenuated replica of a signal that arrives before the actual signal) will be heard in the output. Pre-echo degrades the quality of the dereverberated speech signal. To overcome this problem, a truncated matched filter was suggested in [46.41]. It can reduce but cannot eliminate the pro-echo.

Matched filtering is a computationally efficient way to invert an acoustic channel when its impulse response is known, but it is by no means ideal. It can only marginally improve the signal-to-reverberation ratio and has little conducive impact on reverberation time. Its superiority over the delay-and-sum beamformer is not always observed and is not practically evident.

MINT Method

For a SIMO system, let us consider the polynomials $G_n(z)$ $(n = 1, 2, \cdots, N)$ and the following equation:

$$\begin{aligned} \hat{S}(z) &= \sum_{n=1}^{N} G_n(z) X_n(z) \\ &= \left[\sum_{n=1}^{N} H_n(z) G_n(z)\right] S(z) + \sum_{n=1}^{N} G_n(z) B_n(z)\,. \end{aligned} \tag{46.32}$$

The polynomials $G_n(z)$ should be found in such a way that $\hat{S}(z) = S(z)$ in the absence of noise by using the Bezout theorem, which is mathematically expressed as

$$\begin{aligned} &\gcd[H_1(z), H_2(z), \cdots, H_N(z)] = 1 \\ &\Leftrightarrow \exists G_1(z), G_2(z), \cdots, G_N(z): \\ &\sum_{n=1}^{N} H_n(z) G_n(z) = 1\,, \end{aligned} \tag{46.33}$$

where $\gcd[\cdot]$ denotes the greatest common divisor of the polynomials involved. In other words, as long as the channel impulse responses h_n are co-prime (even though they may not be minimum phase), i. e., the SIMO system is irreducible, there exists a group of g filters to completely remove the reverberations and perfectly recover the source signals. The idea of using the Bezout theorem to equalize a SIMO system was first proposed in [46.26] in the context of room acoustics, where the principle is more widely referred to as the MINT theory.

If the channels of the SIMO system share common zeros, i. e.,

$$C(z) = \gcd[H_1(z), H_2(z), \cdots, H_N(z)] \neq 1\,, \tag{46.34}$$

then we have

$$H_n(z) = C(z) H_n'(z)\,, \quad n = 1, 2, \cdots, N\,, \tag{46.35}$$

and the polynomials $G_n(z)$ can be found such that

$$\sum_{n=1}^{N} H_n'(z) G_n(z) = 1\,. \tag{46.36}$$

In this case, (46.32) becomes

$$\hat{S}(z) = C(z) S(z) + \sum_{n=1}^{N} G_n(z) B_n(z)\,. \tag{46.37}$$

We see that by using the Bezout theorem, the *reducible* SIMO system can be equalized up to the polynomial $C(z)$. So when there are common zeros, the MINT equalizer can only partially suppress the reverberations. For more complete cancelation of the room reverberations, we have to combat the effect of $C(z)$ using either the direct inverse or MMSE/LS methods, which depends on whether $C(z)$ is a minimum-phase filter.

To find the MINT equalization filters, we write the Bezout equation (46.33) in the time domain as

$$\boldsymbol{H}^{\mathrm{c}} \boldsymbol{g} = \sum_{n=1}^{N} \boldsymbol{H}_n^{\mathrm{c}} \boldsymbol{g}_n = \boldsymbol{e}_1\,, \tag{46.38}$$

where

$$\begin{aligned} \boldsymbol{H}^{\mathrm{c}} &= \left[\boldsymbol{H}_1^{\mathrm{c}} \;\; \boldsymbol{H}_2^{\mathrm{c}} \;\; \cdots \;\; \boldsymbol{H}_N^{\mathrm{c}}\right], \\ \boldsymbol{g} &= \left[\boldsymbol{g}_1^{\mathrm{T}} \;\; \boldsymbol{g}_2^{\mathrm{T}} \;\; \cdots \;\; \boldsymbol{g}_N^{\mathrm{T}}\right]^{\mathrm{T}}, \\ \boldsymbol{g}_n &= \left[g_n(0) \;\; g_n(1) \;\; \cdots \;\; g_n(L_g - 1)\right]^{\mathrm{T}}, \end{aligned}$$

L_g is the length of the FIR filter g_n,

$$\boldsymbol{H}_n^{\mathrm{c}} = \begin{pmatrix} h_n(0) & 0 & \cdots & 0 \\ h_n(1) & h_n(0) & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ h_n(L_h-1) & \cdots & \cdots & \vdots \\ 0 & h_n(L_h-1) & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & h_n(L_h-1) \end{pmatrix}$$

is an $(L_h+L_g-1)\times L_g$ convolution matrix (superscript c), and

$$\boldsymbol{e}_1 = \begin{pmatrix} 1 & 0 & \cdots & 0 \end{pmatrix}^{\mathrm{T}}$$

is an $(L_h+L_g-1)\times 1$ vector.

In order to have a unique solution for (46.38), L_g must be chosen in such a way that $\boldsymbol{H}^{\mathrm{c}}$ is a square matrix. In this case, we have:

$$L_g = \frac{L_h-1}{N-1} \,. \tag{46.39}$$

However, this may not be practical since $(L_h-1)/(N-1)$ is not necessarily always an integer. Therefore, a larger L_g is usually chosen and we solve (46.38) for $\boldsymbol{g}$ in the least-squares sense as

$$\hat{\boldsymbol{g}}_{\mathrm{MINT}} = \boldsymbol{H}^{\mathrm{c}\dagger}\boldsymbol{e}_1 \,, \tag{46.40}$$

where

$$\boldsymbol{H}^{\mathrm{c}\dagger} = \left(\boldsymbol{H}^{\mathrm{c}\mathrm{T}}\boldsymbol{H}^{\mathrm{c}}\right)^{-1}\boldsymbol{H}^{\mathrm{c}\mathrm{T}}$$

is the pseudo-inverse of the matrix $\boldsymbol{H}^{\mathrm{c}}$.

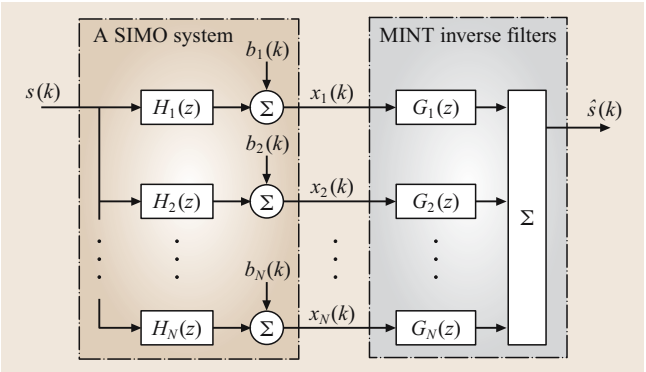


Fig. 46.8 Illustration of the MINT approaches to speech dereverberation for a single-input multiple-output acoustic system

If a decision delay d is taken into account, then the equalization filters turn out to be

$$\hat{\boldsymbol{g}}_{\mathrm{MINT}} = \boldsymbol{H}^{\mathrm{c}\dagger}\boldsymbol{e}_d \,, \tag{46.41}$$

where

$$\boldsymbol{e}_d = \left(\underbrace{0 \cdots 0}_{d} \; 1 \; \underbrace{0 \cdots 0}_{L_h+L_g-d-2}\right)^{\mathrm{T}} .$$

MINT equalization is an appealing approach to speech dereverberation. It can perfectly remove the effect of room reverberation even though acoustic impulse responses are not minimum phase. But in practice, the MINT method is found to be very sensitive to even small errors in the estimated channel impulse responses. Therefore, it is only useful when background noise is weak or well controlled.

46.6 Summary

In this chapter we have attempted to present the state of the art of speech dereverberation techniques. We have explained how room reverberation smears clean speech signals and hence impairs their intelligibility. The difficulty of solving this problem in practice has been elucidated. We have introduced a new classification of speech dereverberation algorithms, according to which well-known and recent developments fall into the following three categories: source model-based speech enhancement algorithms for dereverberation, separation of speech and reverberation via homomorphic transformation, and speech dereverberation by channel inversion and equalization. We have chosen to focus on the ideas behind such a rather large number of approaches for both single-channel and multichannel systems; hence we have avoided tedious derivations of some of the algorithms. We have discussed in details the merits and drawbacks of these speech dereverberation algorithms, trying to shed light on the performance that we believe can be expected when they are implemented in real-world systems.

References

46.1 A.K. Nábělek: Communication in noisy and reverberant environments. In: *Acoustical Factors Affecting Hearing Aid Performance*, 2nd edn., ed. by G.A. Studebaker, I. Hochberg (Allyn Bacon, Needham Height 1993)

46.2 S. Sandhu, O. Ghitza: A comparative study of MEL cepstra and EIH for phone classification under adverse conditions, Proc. IEEE ICASSP **1**, 409–412 (1995)

46.3 B.E.D. Kingsbury, N. Morgan: Recognizing reverberant speech with RASTA-PLP, Proc. IEEE ICASSP **2**, 1259–1262 (1997)

46.4 D.A. Berkley, J.B. Allen: Normal listening in typical rooms – the physical and psychophysical correlates of reverberation. In: *Acoustical Factors Affecting Hearing Aid Performance*, 2nd edn., ed. by G.A. Studebaker, I. Hochberg (Allyn Bacon, Needham Height 1993)

46.5 R.Y. Litovsky, H.S. Colburn, W.A. Yost, S.J. Guzman: The precedence effect, J. Acoust. Soc. Am. **106**, 1633–1654 (1999)

46.6 W.C. Sabine: *Collected Papers on Acoustics* (Harvard Univ. Press, Cambridge 1922)

46.7 M.R. Schroeder: New method of measuring reverberation time, J. Acoust. Soc. Am. **37**, 409–412 (1965)

46.8 D.R. Morgan: A parametric error analysis of the backward integration method for reverberation time estimation, J. Acoust. Soc. Am. **101**, 2686–2693 (1997)

46.9 H. Kuttruff: *Room Acoustics*, 4th edn. (Spon, New York 2000)

46.10 M.S. Brandstein, S.M. Griebel: Nonlinear, model-based microphone array speech enhancement. In: *Acoustic Signal Processing for Telecommunication*, ed. by S.L. Gay, J. Benesty (Kluwer Academic, Boston 2000) pp. 261–279, Chap. 12

46.11 B. Yegnanarayana, P.S. Murthy: Enhancement of reverberant speech using LP residual signal, IEEE Trans. Speech Audio Process. **8**, 267–281 (2000)

46.12 B. Yegnanarayana, S.R. Mahadeva Prasanna, K. Sreenivasa Rao: Speech enhancement using excitation source information, Proc. IEEE ICASSP **1**, 541–544 (2002)

46.13 B.W. Gillespie, H.S. Malvar, D.A.F. Florencio: Speech dereveration via maximum-kurtosis subband adaptive filtering, Proc. IEEE ICASSP **6**, 3701–3704 (2001)

46.14 T. Nakatani, M. Miyoshi, K. Kinoshita: Single-microphone blind dereverberation. In: *Speech Enhancement*, ed. by J. Benesty, S. Makino, J. Chen (Springer, Berlin, Heidelberg 2005) pp. 247–270, Chap. 11

46.15 H. Attias, J.C. Platt, A. Acero, L. Deng: Speech denoising and dereverberation using probabilistic models. In: *Advances in Neural Information Processing Systems*, Vol. 13, ed. by S. Thrun, L. Saul, B. Schölkopf (MIT Press, Cambridge 2001) pp. 758–764

46.16 A.V. Oppenheim, R.W. Schafer: *Discrete-Time Signal Processing* (Prentice-Hall, Englewood Cliffs 1989)

46.17 D. Bees, M. Blostein, P. Kabal: Reverberant speech enhancement using cepstral processing, Proc. IEEE ICASSP **2**, 977–980 (1991)

46.18 M. Tohyama, R.H. Lyon, T. Koike: Source waveform recovery in a reverberant space by cepstrum dereverberation, Proc. IEEE ICASSP **1**, 157–160 (1993)

46.19 B.-H. Juang, L.R. Rabiner, J.G. Wilpon: On the use of bandpass liftering in speech recognition, Proc. IEEE ICASSP **1**, 765–768 (1986)

46.20 Y. Tohkura: A weighted cepstral distance measure for speech recognition, IEEE Trans. Acoust. Speech Signal Process. **ASSP-35**(10), 1414–1422 (1987)

46.21 B.S. Atal: Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification, J. Acoust. Soc. Am. **55**(6), 1304–1312 (1974)

46.22 H. Hermansky, N. Morgan: RASTA processing of speech, IEEE Trans. Speech Audio Process. **2**(4), 578–589 (1994)

46.23 M.G. Rahim, B.-H. Juang: Signal bias removal by maximum likelihood estimation for robust telephone speech recognition, IEEE Trans. Speech Audio Process. **4**(1), 19–30 (1996)

46.24 J.L. Flanagan, J.D. Johnston, R. Zahn, G.W. Elko: Computer steered imcrophone arrays for sound transduction in large rooms, J. Acoust. Soc. Am. **78**(5), 1508–1518 (1985)

46.25 J.L. Flanagan, A.C. Surendran, E.E. Jan: Spatially selective sound capture for speech and audio processing, Speech Commun. **13**, 207–222 (1993)

46.26 M. Miyoshi, Y. Kaneda: Inverse filtering of room acoustics, IEEE Trans. Acoust. Speech Signal Process. **36**, 145–152 (1988)

46.27 Y. Huang, J. Benesty, J. Chen: *Acoustic MIMO Signal Processing* (Springer, Berlin, Heidelberg 2006)

46.28 H. Dudley: The vocoder, Bell Labs Rec. **17**, 122–126 (1939)

46.29 B.S. Atal, M.R. Schroeder: Predictive coding of speech, Proc. AFCRL/IEEE Conf. Speech Communication and Processing (1967) pp. 360–361

46.30 B.S. Atal, M.R. Schroeder: Adaptive predictive coding of speech, Bell Syst. Tech. J. **49**(8), 1973–1986 (1970)

46.31 B.-H. Juang, M.M. Sondhi, L.R. Rabiner: Digital speech processing. In: *Encyclopedia of Physical Science and Technology*, Vol. 4, 3rd edn., ed. by R.A. Meyers (Academic, Orlando 2000) pp. 485–500

46.32 W.J. Hess: Pitch and voicing determination. In: *Advances in Speech Signal Processing*, ed. by S. Fu-

rui, M.M. Sondhi (Marcel Dekker, New York 1992) pp. 3–48
46.33 S.T. Neely, J.B. Allen: Invertibility of a room impulse response, J. Acoust. Soc. Am. **68**, 165–169 (1979)
46.34 O.L. Frost: An algorithm for linearly constrained adaptive array processing, Proc. IEEE **60**, 926–935 (1972)
46.35 L.J. Griffiths, C.W. Jim: An alternative approach to linearly constrained adaptive beamforming, IEEE Trans. Antennas Propagat. **AP-30**, 27–34 (1982)
46.36 B.D. Van Veen, K.M. Buckley: Beamforming: a versatile approach to spatial filtering, IEEE ASSP Mag. **5**, 4–24 (1988)
46.37 S. Affes, S. Gazor, Y. Grenier: Robust adaptive beamforming via LMS-like target tracking, Proc. IEEE ICASSP **4**, 269–272 (1994)
46.38 O. Hoshuyama, A. Sugiyama, A. Hirano: A robust adaptive beamformer for microphone arrays with a blocking matrix using constrained adaptive filter, IEEE Trans. Signal Process. **47**, 2677–2684 (1999)
46.39 S. Gannot, D. Burshtein, E. Weinstein: Signal enhancement using beamforming and nonstationarity with applications to speech, IEEE Trans. Signal Process. **49**, 1614–1626 (2001)
46.40 W. Herbordt, W. Kellermann: Adaptive beamforming for audio signal acquisition. In: *Adaptive Signal Processing: Applications to Real-World Problems*, ed. by J. Benesty, Y. Huang (Springer, Berlin, Heidelberg 2003) pp. 155–194, Chap. 6
46.41 D. Rabinkin, R. Renomeron, J.L. Flanagan, D.F. Macomber: Optimal truncation time for matched filter array processing, Proc. IEEE ICASSP **6**, 3629–3632 (1998)

48. Feedback Control in Hearing Aids

A. Spriet, S. Doclo, M. Moonen, J. Wouters

Acoustic feedback limits the maximum amplification that can be used in a hearing aid without making it unstable. This chapter gives an overview of existing techniques for feedback suppression and, in particular, adaptive feedback cancellation. Because of the presence of a closed signal loop, standard adaptive filtering techniques for open-loop systems fail to provide a reliable feedback path estimate if the desired signal is spectrally colored. Several approaches for improving the estimation accuracy of the adaptive feedback canceller will be reviewed and evaluated for acoustic feedback paths measured in a commercial behind-the-ear hearing aid.

This chapter is organized as follows. Section 48.1 gives a mathematical formulation of the acoustic feedback problem in hearing aids and briefly describes the two possible approaches to reduce its negative effects, i. e., feedforward suppression and feedback cancellation. In addition, performance measures for feedback cancellation are defined.

Section 48.2 discusses the standard continuous adaptation feedback (CAF) cancellation algorithm that is widely studied for application in hearing aids. We demonstrate that the standard CAF suffers from a model error or bias when the desired signal is spectrally colored (e.g., a speech signal). In the literature, several solutions have been proposed to reduce the bias of the CAF. A common approach is to incorporate signal decorrelating operations (such as delays) in the signal processing path of the hearing aid or to reduce the adaptation speed of the adaptive feedback canceller. Other techniques, discussed in Sect. 48.3, exploit prior knowledge of the acoustic feedback path to improve the adaptation of the feedback canceller. In Sect. 48.4, a final class of techniques is presented that view the feedback path as a part of a closed-loop system and apply closed-loop system identification theory [48.1]. Among the different closed-loop identification methods, especially the direct method is an appealing approach for feedback cancellation. In contrast to the other methods, this technique does not require the use of an external probe signal. The direct method reduces the bias of the feedback canceller by incorporating a (stationary or time-varying) model of the desired signal $x[k]$ in the identification.

Finally, Sect. 48.5 compares the steady-state performance as well as the tracking performance of different algorithms for acoustic feedback paths measured in a commercial behind-the-ear hearing aid.

Acoustic feedback in hearing aids refers to the acoustical coupling between the loudspeaker and the microphone of the hearing aid. Because of this coupling, the hearing aid produces a severe distortion of the desired signal and an annoying howling sound when the gain is increased. With the growing use of open fittings, i. e., without an earmold, and the decreasing distance between the loudspeaker and the microphone of the hearing aid, the demand for effective feedback suppression or cancellation techniques increases.

This chapter gives an overview of existing techniques for feedback suppression and, in particular, adaptive feedback cancellation. Because of the presence of a closed signal loop, standard adaptive filtering techniques for open-loop systems (as used in echo cancellation) fail to provide a reliable feedback path estimate if the desired signal is spectrally colored. Several approaches for improving the estimation accuracy of the adaptive feedback canceller will be discussed.

48.1 Problem Statement

Section 48.1.1 defines the notation and gives a mathematical formulation of the acoustic feedback problem in hearing aids. Section 48.1.2 describes the two approaches that can be used to reduce acoustic feedback, i. e., feedforward suppression and feedback cancellation. In Sect. 48.1.3, we define the performance measures that will be used to evaluate the feedback cancellation techniques.

48.1.1 Acoustic Feedback in Hearing Aids

Notation

The superscripts T and H denote matrix/vector transpose and complex conjugate transpose, respectively. The expectation operator is denoted by $\mathcal{E}\{\cdot\}$. The matrix $\mathcal{F}$ is the $M \times M$ discrete Fourier transform (DFT) matrix. The matrix $\mathbf{I}_L$ is the $L \times L$ identity matrix; the matrix $\mathbf{0}_L$ is an $L \times L$ dimensional matrix of zeros.

The discrete-time index is denoted by k. The symbol q^{-1} denotes the discrete-time delay operator, i. e.,

$$q^{-1}u[k] = u[k-1]\,. \tag{48.1}$$

A discrete-time filter with coefficient vector $\boldsymbol{f}[k]$

$$\boldsymbol{f}[k] = \big[f_0[k] \;\; f_1[k] \;\; \cdots \;\; f_{L_F-1}[k]\big]^{\mathrm{T}} \tag{48.2}$$

and filter length L_F is represented as a polynomial transfer function $F(q,k)$ in q, i. e.,

$$F(q,k) = \boldsymbol{f}^{\mathrm{T}}[k]\boldsymbol{q}\,, \tag{48.3}$$

with $\boldsymbol{q} = \big[1 \;\; q^{-1} \;\; \ldots \;\; q^{-L_F+1}\big]^{\mathrm{T}}$. This representation, which is adopted from [48.2], allows the following notation for the filtering of $u[k]$ with $F(q,k)$:

$$F(q,k)u[k] = \boldsymbol{f}^{\mathrm{T}}[k]\boldsymbol{u}[k]\,, \tag{48.4}$$

with

$$\boldsymbol{u}[k] = \big[u[k] \;\; u[k-1] \;\; \cdots \;\; u[k-L_F+1]\big]^{\mathrm{T}}\,. \tag{48.5}$$

The spectrum of $F(q,k)$ is denoted by $F(\mathrm{e}^{\mathrm{i}\omega},k)$ with $\omega \in [0,\pi]$ being the normalized angular frequency.

Acoustic Feedback

Figure 48.1 illustrates the problem of acoustic feedback for a hearing aid with a single microphone. The so-called *forward path* $G(q,k)$ represents the regular signal processing path of the hearing aid, typically including a frequency-specific gain, compression, and/or noise reduction. We assume that $G(q,k)$ has a delay d_G of at least one sample, i. e., $g_0 = 0$.

The *feedback path* between the loudspeaker and the microphone is denoted by $F(q,k)$. It includes the frequency characteristic of the loudspeaker and the microphone, the acoustic feedback path, and the characteristics of the digital-to-analog (DAC) and analog-to-digital (ADC) converters. The feedback path $F(q,k)$ typically contains a delay d_F that arises from the processing delay of the ADC and DAC convert-

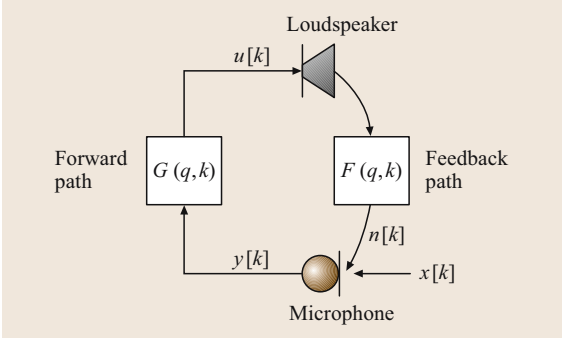


Fig. 48.1 Acoustic feedback

ers and the propagation time of the sound from the loudspeaker to the microphone. For simplicity, we will include this delay d_F in the delay d_G of the forward path $G(q,k)$. The loudspeaker and microphone signal are $u[k]$ and $y[k]$, respectively. The desired sound signal (i. e., without acoustic feedback) is denoted by $x[k]$ and the feedback signal is denoted by $n[k] = F(q,k)u[k]$.

Ideally, the loudspeaker signal $u[k]$ equals the desired signal $x[k]$ processed by the forward path $G(q,k)$, i. e.,

$$u[k] = G(q,k)x[k] \,. \tag{48.6}$$

However, because of acoustic feedback, the amplified sound $u[k]$ sent through the loudspeaker is fed back into the microphone, resulting in a closed-loop system $C(q,k)$

$$C(q,k) = \frac{G(q,k)}{1 - G(q,k)F(q,k)} \tag{48.7}$$

from $x[k]$ to the loudspeaker signal $u[k]$.

Instability occurs if the loop gain $|G(\mathrm{e}^{\mathrm{i}\omega},k)F(\mathrm{e}^{\mathrm{i}\omega},k)|$ exceeds one at an angular frequency $\omega \in [0,\pi]$ with positive feedback, i. e., an angular frequency where the loop phase $\angle(G(\mathrm{e}^{\mathrm{i}\omega},k)F(\mathrm{e}^{\mathrm{i}\omega},k))$ equals a multiple of 2π. If the gain $|G(\mathrm{e}^{\mathrm{i}\omega},k)|$ is increased beyond this point, the system starts to oscillate. However also at low gain margins, acoustic feedback already degrades the sound quality by introducing ringing or howling. To avoid significant audible distortion, a gain margin of at least 6 dB is advisable: the effect of the acoustic feedback on the closed-loop frequency response $C(\mathrm{e}^{\mathrm{i}\omega},k)$ is then limited to [48.3]

$$\frac{2}{3}G(\mathrm{e}^{\mathrm{i}\omega},k) \le C(\mathrm{e}^{\mathrm{i}\omega},k) \le 2G(\mathrm{e}^{\mathrm{i}\omega},k). \tag{48.8}$$

As an illustration, Fig. 48.2 depicts the feedback path frequency response measured in a commercial (GN ReSound) behind-the-ear hearing aid. During the measurement, the hearing aid was attached to the right ear of a Cortex MK II artificial head. An earmold with a vent size of 2 mm was used. In this example, the minimum feedback path attenuation, and hence, the maximum amplification without instability lies around 20 dB. The acoustic feedback path when a mobile phone is attached to the head is also shown. This demonstrates that placing a telephone handset close to the ear can temporarily reduce the feedback path attenuation by another 10–20 dB [48.4–6].

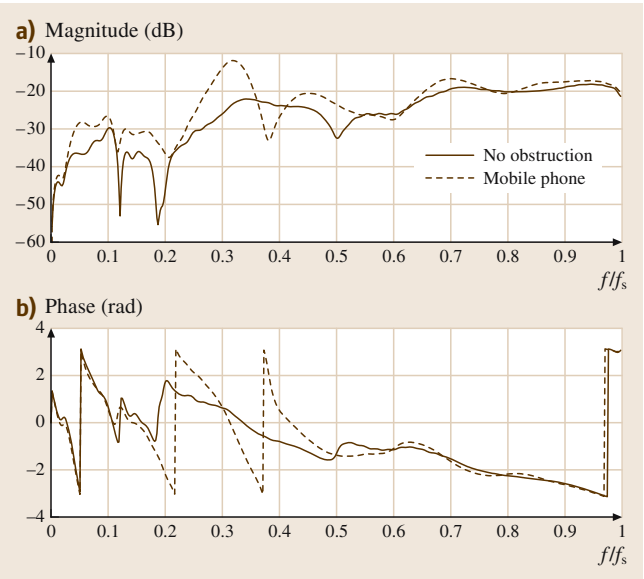


Fig. 48.2 Feedback path frequency response of a commercial, GN ReSound behind-the-ear hearing aid connected to a vented earmold (with a vent size of 2 mm) for two conditions, i. e., without obstruction and with a mobile phone attached to the ear

48.1.2 Feedforward Suppression Versus Feedback Cancellation

To reduce the negative effects introduced by acoustic feedback, several techniques have been proposed in the literature. They can be broadly classified into feedforward suppression and feedback cancellation techniques.

Feedforward Suppression

In *feedforward suppression* techniques, the regular signal processing path $G(q,k)$ of the hearing aid is modified in such a way that it is stable in conjunction with the feedback path $F(q,k)$. The most common technique is the use of a notch filter. In a notch filter, the gain is reduced in a narrow frequency band around the critical frequencies, whenever feedback occurs [48.7–10]. The main disadvantage is that this approach is reactive: in order to identify the oscillation frequencies, howling first has to occur. Other examples include equalizing the phase of the open-loop response [48.11] and using time-varying elements (such as frequency shifting, delay and phase modulation) in the forward path $G(q,k)$ [48.7, 12, 13].

The achievable increase in maximum stable gain with feedforward suppression techniques was generally found to be limited [48.7, 12]. In addition, feedforward suppression techniques all compromise the basic fre-

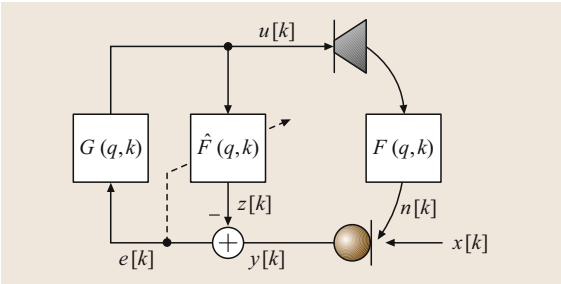


Fig. 48.3 Adaptive feedback cancellation

quency response of the hearing aid, and hence, may seriously affect the sound quality.

Feedback Cancellation

A more-promising solution for acoustic feedback is the use of a *feedback cancellation* algorithm, which is illustrated in Fig. 48.3. The feedback canceller $\hat{F}(q,k)$ produces an estimate $z[k]$ of the feedback signal $n[k]$ and subtracts this estimate $z[k]$ from the microphone signal $y[k]$. As a result, the closed-loop transfer function $C(q,k)$ from the desired signal $x[k]$ to the loudspeaker signal $u[k]$ is modified to

$$C(q,k) = \frac{G(q,k)}{1 - G(q,k)[F(q,k) - \hat{F}(q,k)]} \,. \qquad (48.9)$$

Ideally, $\hat{F}(q,k) = F(q,k)$ such that the closed-loop system $C(q,k)$ equals the desired hearing aid response $G(q,k)$.

The acoustic path $F(q,k)$ between the loudspeaker and the microphone can vary significantly depending on the acoustical environment (see Fig. 48.2) [48.4–6, 14]. Hence, adaptive feedback cancellers $\hat{F}(q,k)$ are called for. In the remainder of this chapter, existing adaptive feedback cancellation techniques will be discussed.

48.1.3 Performance of a Feedback Canceller

The most common measures to assess the performance of feedback cancellation algorithms are the misalignment between the true and estimated feedback path $\hat{F}(q,k)$ and the maximum stable gain [48.14–18].

The *misalignment* $\zeta_k(F,\hat{F})$ between the true feedback path $F(q,k)$ and the feedback path estimate $\hat{F}(q,k)$ is computed in the frequency domain as

$$\zeta_k(F,\hat{F}) = \sqrt{\frac{\int_0^\pi |F(\mathrm{e}^{\mathrm{i}\omega},k) - \hat{F}(\mathrm{e}^{\mathrm{i}\omega},k)|^2 \,\mathrm{d}\omega}{\int_0^\pi |F(\mathrm{e}^{\mathrm{i}\omega},k)|^2 \,\mathrm{d}\omega}} \,. \qquad (48.10)$$

The misalignment reflects the accuracy of the feedback path estimate $\hat{F}(q,k)$.

To get an idea of the actual benefit of a feedback cancellation algorithm for the hearing aid user, the *maximum stable gain* (MSG) is often used [48.14,16]. In this chapter, the MSG is defined as the largest gain for which the hearing aid does not oscillate and the desired signal is not substantially degraded by algorithm artifacts (e.g., signal distortion or howling) [48.16]. The MSG is determined by gradually and slowly increasing the gain while the feedback canceller is running.

48.2 Standard Adaptive Feedback Canceller

Adaptive feedback cancellers can be divided into two classes: algorithms with a continuous adaptation and algorithms with a *noncontinuous adaptation* [48.9,16,19]. The latter only adapt the coefficients of the feedback canceller when instability is detected or when the input signal level is low [48.19–21]. Due to this reactive, rather than proactive, adaptation, such systems may be objectionable. A *continuous adaptation* feedback canceller, on the other hand, continuously adapts the coefficients of the feedback canceller.

This section describes a standard continuous adaptation feedback (CAF) cancellation technique that is widely studied for hearing-aid applications [48.9,16,22]. In Sect. 48.2.2, we show that, because of the presence of a closed signal loop, the standard CAF suffers from a large model error or bias if the desired signal is spectrally colored. Classical approaches for reducing the bias of the CAF are discussed in Sect. 48.2.3.

48.2.1 Adaptation of the CAF

The CAF continuously adapts the coefficients

$$\hat{\boldsymbol{f}}[k] = \left[f_0[k] \;\; f_1[k] \;\; \cdots \;\; f_{L_{\hat{F}}-1}[k] \right]^{\mathrm{T}} \qquad (48.11)$$

of the feedback canceller based on standard adaptive filtering (Wiener filtering) procedures (Fig. 48.3). The CAF minimizes the energy of the feedback compensated

signal $e[k]$, i. e.,

$$J(\hat{\boldsymbol{f}}[k]) = \mathcal{E}\{|y[k] - \hat{\boldsymbol{f}}^{\mathrm{T}}[k]\boldsymbol{u}[k]|^2\}\,, \tag{48.12}$$

with

$$\boldsymbol{u}[k] = \left[u[k] \;\; \cdots \;\; u[k - L_{\hat{F}} + 1]\right]^{\mathrm{T}}, \tag{48.13}$$

resulting in the well-known Wiener filter

$$\hat{\boldsymbol{f}}[k] = \mathcal{E}\{\boldsymbol{u}[k]\boldsymbol{u}^{\mathrm{T}}[k]\}^{-1}\mathcal{E}\{\boldsymbol{u}[k]y[k]\}\,. \tag{48.14}$$

Assuming the sufficient-order case, where $L_{\hat{F}} = L_F$, and using

$$y[k] = \boldsymbol{f}^{\mathrm{T}}[k]\boldsymbol{u}[k] + x[k], \tag{48.15}$$

(48.14) can be written as:

$$\begin{aligned}\hat{\boldsymbol{f}}[k] &= \mathcal{E}\{\boldsymbol{u}[k]\boldsymbol{u}^{\mathrm{T}}[k]\}^{-1}\mathcal{E}\{\boldsymbol{u}[k]\boldsymbol{u}^{\mathrm{T}}[k]\boldsymbol{f}[k]\} \\ &\quad + \mathcal{E}\{\boldsymbol{u}[k]\boldsymbol{u}^{\mathrm{T}}[k]\}^{-1}\mathcal{E}\{\boldsymbol{u}[k]x[k]\}. \\ &= \boldsymbol{f}[k] + \mathcal{E}\{\boldsymbol{u}[k]\boldsymbol{u}^{\mathrm{T}}[k]\}^{-1}\mathcal{E}\{\boldsymbol{u}[k]x[k]\}\,. \end{aligned} \tag{48.16}$$

From (48.16), it can be seen that the desired signal $x[k]$ acts as a disturbance to the feedback canceller.

Standard adaptive filter procedures (e.g., least mean squares (LMS), recursive least squares (RLS)) can be used for adapting the filter coefficients $\hat{\boldsymbol{f}}[k]$.

Time-Domain Normalized LMS

The most popular adaptive filtering technique is the NLMS algorithm, i. e.,

$$\begin{aligned}\hat{\boldsymbol{f}}[k] &= \hat{\boldsymbol{f}}[k-1] + \mu[k]\boldsymbol{u}[k] \\ &\quad \times (y[k] - \hat{\boldsymbol{f}}^{\mathrm{T}}[k-1]\boldsymbol{u}[k])\,. \end{aligned} \tag{48.17}$$

Unlike in acoustic echo cancellation, the feedback canceller adapts during a continuous double-talk situation [see (48.16)]. To reduce the excess mean square error (MSE) in the presence of desired signals $x[k]$ with large power fluctuations or signal onsets, the step size $\mu[k]$ of the NLMS algorithm is normalized with the sum of the short-term average input power $\overline{|u[k]|^2}$ and error power $\overline{|e[k]|^2}$ [48.16, 23]:

$$\mu[k] = \frac{\bar{\mu}}{L_{\hat{F}}\left(\overline{|u[k]|^2} + \overline{|e[k]|^2}\right) + c}\,, \tag{48.18}$$

where $\bar{\mu}$ is a dimensionless step size constant, c is a small positive constant to prevent singularities, and

$$\overline{|u[k]|^2} = (1-\gamma)|u[k]|^2 + \gamma\overline{|u[k-1]|^2}\,, \tag{48.19}$$

$$\overline{|e[k]|^2} = (1-\gamma)|e[k]|^2 + \gamma\overline{|e[k-1]|^2}\,, \tag{48.20}$$

with $\gamma \in (0, 1]$. For a highly time-varying signal $x[k]$ such as a speech signal, a burst in $e[k]$ most often originates from an increase in the short-term power of $x[k]$. Hence, normalization with the error power reduces the step size $\mu[k]$ when the desired signal $x[k]$ is strong and thus mitigates the negative effect of a strong desired signal segment on the excess MSE.

Partitioned-Block Frequency-Domain NLMS

Better performance can be achieved when updating the feedback canceller in the frequency domain [48.24–27]. Partitioned-block frequency-domain (PBFD) adaptive filters combine the beneficial properties of frequency-domain algorithms, i. e., faster convergence (thanks to a frequency-dependent step size control) and a reduced complexity, with a small processing delay as required for hearing aids.

In this chapter, we concentrate on a PBFD implementation based on overlap-save [48.27]. In this implementation, the $L_{\hat{F}}$-taps feedback canceller $\hat{\boldsymbol{f}}[k]$ is partitioned into $L_{\hat{F}}/P$ segments $\hat{\boldsymbol{f}}_p[k]$ of length P each. Each segment $\hat{\boldsymbol{f}}_p[k]$, $p = 0, \ldots, L_{\hat{F}}/P - 1$ is then transformed to the frequency-domain vector $\hat{\boldsymbol{F}}_p[k]$ by means of the M-point DFT matrix $\mathcal{F}$:

$$\hat{\boldsymbol{f}}_p[k] = \left[\hat{f}_{pP}[k] \;\; \ldots \;\; \hat{f}_{(p+1)P-1}[k]\right]^{\mathrm{T}}, \tag{48.21}$$

$$\hat{\boldsymbol{F}}_p[k] = \mathcal{F}\begin{bmatrix}\hat{\boldsymbol{f}}_p[k] \\ \boldsymbol{0}\end{bmatrix}\begin{matrix}\updownarrow P \\ \updownarrow M-P\end{matrix}. \tag{48.22}$$

(It is assumed that $L_{\hat{F}}/P \in \mathbb{N}_0$. Otherwise $\hat{\boldsymbol{f}}_p[k]$ has to be padded with zeros.) Let the L-dimensional block signal $\boldsymbol{u}_{\bar{k}}$ be defined as

$$\boldsymbol{u}_{\bar{k}} = \left[u[\bar{k}L+1] \;\; \ldots \;\; u[(\bar{k}+1)L]\right]^{\mathrm{T}}, \tag{48.23}$$

with $\bar{k}$ the block index. For each block of L input samples $\boldsymbol{u}_{\bar{k}}$, the PBFD NLMS filter produces L output samples $\boldsymbol{z}_{\bar{k}} = [z[\bar{k}L+1] \; \ldots \; z[(\bar{k}+1)L]]^{\mathrm{T}}$:

$$\boldsymbol{U}_p[\bar{k}] = \mathrm{diag}\left\{\mathcal{F}\begin{bmatrix}u[(\bar{k}+1)L - pP - M + 1] \\ \vdots \\ u[(\bar{k}+1)L - pP]\end{bmatrix}\right\}, \tag{48.24}$$

$$\boldsymbol{z}_{\bar{k}} = \begin{bmatrix}\boldsymbol{0} & \mathbf{I}_L\end{bmatrix}\mathcal{F}^{-1}\sum_{p=0}^{\frac{L_{\hat{F}}}{P}-1}\boldsymbol{U}_p[\bar{k}]\hat{\boldsymbol{F}}_p[\bar{k}]\,. \tag{48.25}$$

The parameter L is called the block length and hence the corresponding input/output delay of the PBFD implementation equals $2L - 1$. To ensure proper operation, it is required that the DFT length $M \geq P + L - 1$.

The PBFD update equation is

$$\hat{\boldsymbol{F}}_p[\bar{k}+1] = \hat{\boldsymbol{F}}_p[\bar{k}] + \boldsymbol{\Delta}[\bar{k}]\mathcal{F}\boldsymbol{g}\mathcal{F}^{-1}\boldsymbol{U}_p^{\mathrm{H}}[\bar{k}]\boldsymbol{E}[\bar{k}]\,, \tag{48.26}$$

where

$$\boldsymbol{E}[\bar{k}] = \mathcal{F}\begin{bmatrix}\boldsymbol{0}\\ \mathbf{I}_L\end{bmatrix}(\boldsymbol{y}_{\bar{k}} - \boldsymbol{z}_{\bar{k}})\,, \tag{48.27}$$

$$\boldsymbol{y}_{\bar{k}} = \big[y[\bar{k}L+1] \;\; \ldots \;\; y[(\bar{k}+1)L]\big]^{\mathrm{T}}\,, \tag{48.28}$$

and

$$\boldsymbol{g} = \begin{bmatrix}\mathbf{I}_P & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0}_{M-P}\end{bmatrix}. \tag{48.29}$$

The matrix $\boldsymbol{\Delta}[\bar{k}]$ in (48.26) is a diagonal matrix that contains the step sizes $\mu_m[\bar{k}]$, $m = 0, \ldots, M-1$ of the different frequency bins m, i. e.,

$$\boldsymbol{\Delta}[\bar{k}] = \mathrm{diag}\big\{\mu_0[\bar{k}], \;\; \ldots \;\; \mu_{M-1}[\bar{k}]\big\}. \tag{48.30}$$

Each step size $\mu_m[\bar{k}]$ is normalized according to the sum of the input and the error power in each frequency bin m:

$$\mu_m[\bar{k}] = \frac{\bar{\mu}_m}{\sum_{p=0}^{\frac{L_{\hat{F}}}{P}-1}|U_{p,m}[\bar{k}]|^2 + \frac{L_{\hat{F}}}{P}|E_{p,m}[\bar{k}]|^2 + c}\,, \tag{48.31}$$

where $U_{p,m}[\bar{k}]$ and $E_{p,m}[\bar{k}]$ are the m-th element of $\boldsymbol{U}_p[\bar{k}]$ and $\boldsymbol{E}_p[\bar{k}]$, respectively, and c is a small constant.

The update equations of the PBFD $\hat{\boldsymbol{F}}_p[\bar{k}]$ are summarized in Table 48.1. In the simulations, we set $M = 64$, $P = 32$, and $\bar{\mu}_m = 0.01$. The block length L is chosen such that the processing delay $2L-1$ does not exceed the desired total processing delay d_G (e.g., $L = 32$ for $d_G \geq 4$ ms; $L = 8$ for $d_G = 1$ ms; $L = 16$ for $d_G = 2$–3 ms).

48.2.2 Bias of the CAF

The presence of the closed signal loop $G(q, k)$ introduces specific signal correlation when the desired signal $x[k]$ is spectrally colored (e.g., a speech or music signal). As a result, the CAF fails to provide a reliable feedback path estimate [48.28].

Let us again assume the sufficient-order case where $L_{\hat{F}} = L_F$. Then, the feedback path estimate $\hat{\boldsymbol{f}}[k]$ can be decomposed as [see (48.16)]:

$$\hat{\boldsymbol{f}}[k] = \boldsymbol{f}[k] + \underbrace{\mathcal{E}\{\boldsymbol{u}[k]\boldsymbol{u}^{\mathrm{T}}[k]\}^{-1}\mathcal{E}\{\boldsymbol{u}[k]x[k]\}}_{\text{bias}}\,. \tag{48.32}$$

If $\mathcal{E}\{\boldsymbol{u}[k]x[k]\} = \boldsymbol{0}$, the feedback path estimate $\hat{\boldsymbol{f}}[k]$ is unbiased.

However, because of the presence of the forward path $G(q, k)$, the input signal $u[k]$ to the adaptive filter

Table 48.1 PBFD implementation of CAF

For each block of L input samples $[u[\bar{k}L+1], \ldots, u[(\bar{k}+1)L]$:
$\forall p = 0, \ldots, \frac{L_{\hat{F}}}{P} - 1$:
$\mathbf{U}_p[\bar{k}] = \mathrm{diag}\left\{\mathcal{F}\begin{bmatrix}u[(\bar{k}+1)L - pP - M + 1]\\ \vdots \\ u[(\bar{k}+1)L - pP]\end{bmatrix}\right\}$.
Block of output samples $\boldsymbol{z}_{\bar{k}} = [z[\bar{k}L+1] \ldots z[(\bar{k}+1)L]]^{\mathrm{T}}$:
$\boldsymbol{z}_{\bar{k}} = [\boldsymbol{0} \;\; \boldsymbol{I}_L]\mathcal{F}^{-1}\sum_{p=0}^{\frac{L_{\hat{F}}}{P}-1}\boldsymbol{U}_p[\bar{k}]\hat{\boldsymbol{F}}_p[\bar{k}]$.
Update formula:
$\boldsymbol{y}_{\bar{k}} = [y[\bar{k}L+1] \;\; \ldots \;\; y[(\bar{k}+1)L]]^{\mathrm{T}}$,
$\boldsymbol{E}[\bar{k}] = \mathcal{F}\begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{I}_L\end{bmatrix}\big(\boldsymbol{y}_{\bar{k}} - [\boldsymbol{0} \;\; \boldsymbol{I}_L]\mathcal{F}^{-1}\sum_{p=0}^{\frac{L_{\hat{F}}}{P}-1}\boldsymbol{U}_p[\bar{k}]\hat{\boldsymbol{F}}_p[\bar{k}]\big)$,
$\hat{\boldsymbol{F}}_p[\bar{k}+1] = \hat{\boldsymbol{F}}_p[\bar{k}] + \boldsymbol{\Delta}[\bar{k}]\mathcal{F}\boldsymbol{g}\mathcal{F}^{-1}\boldsymbol{U}_p^{\mathrm{H}}[\bar{k}]\boldsymbol{E}[\bar{k}]$.
Step size:
$\boldsymbol{\Delta}[\bar{k}] = \mathrm{diag}\{\mu_0[\bar{k}], \ldots, \mu_{M-1}[\bar{k}]\}$,
$\mu_m[\bar{k}] = \frac{\bar{\mu}_m}{\sum_{p=0}^{\frac{L_{\hat{F}}}{P}-1}|U_{p,m}[\bar{k}]|^2 + \frac{L_{\hat{F}}}{P}|E_{p,m}[\bar{k}]|^2 + c}$.

$\hat{\boldsymbol{f}}[k]$ relates to $x[k]$ as

$$u[k] = C(q,k)x[k]\,, \tag{48.33}$$

where $C(q,k)$ is defined in (48.9).

Usually, $G(q,k)$ contains a processing delay d_G of at least one sample. (Note that the delay d_G includes the processing delay of the DAC and the ADC of the hearing aid (Sect. 48.1).) Assuming that $G(q,k)$, $F(q,k)$, and $\hat{F}(q,k)$ are causal, the closed-loop system $C(q,k)$ can be specified as

$$\begin{aligned} C(q,k) = {} & c_{d_G}[k]q^{-d_G} + c_{d_G+1}[k]q^{-d_G-1} \\ & + \cdots + c_{L_C-1}[k]q^{-L_C+1} \end{aligned} \tag{48.34}$$

and hence

$$\begin{aligned} u[k] = {} & \big(c_{d_G}[k] + \ldots \\ & + c_{L_C-1}[k]q^{-L_C+d_G+1}\big)x[k-d_G]\,. \end{aligned} \tag{48.35}$$

As a result, $\mathcal{E}\{\boldsymbol{u}[k]x[k]\} = \mathbf{0}$ if and only if

$$\mathcal{E}\{x[k-\delta]x[k]\} = 0 \tag{48.36}$$

for $d_G \le \delta \le L_C + L_{\hat{F}} - 2$.

Most practical sound signals $x[k]$ are *spectrally colored*, meaning that the signal values $x[k]$ are correlated in time (e.g., speech, music, etc.). Many of these audio signals may be well approximated as low-order time-varying autoregressive (AR) random processes

$$x[k] = H(q,k)w[k] = \frac{1}{A(q,k)}w[k]\,, \tag{48.37}$$

where $A(q,k)$ is an finite impulse response (FIR) with $a_0 = 1$ and where $w[k]$ is a white noise signal. Hence, the signal model $H(q)$ is often infinite impulse response (IIR), so that the length L_H of $H(q)$ typically exceeds d_G and hence, $\mathcal{E}\{\boldsymbol{u}[k]x[k]\} \neq 0$. The CAF will then cancel the desired signal $x[k]$ instead of the feedback signal $n[k] = F(q,k)u[k]$ [see (48.16)], leading to signal distortion.

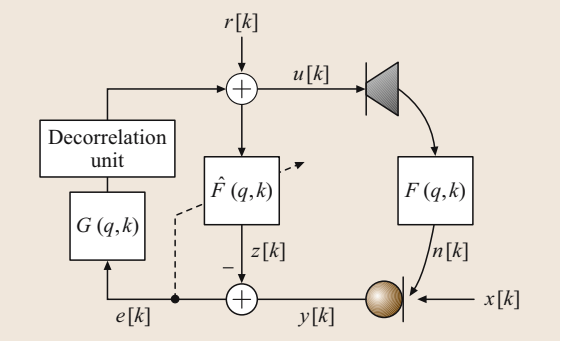


Fig. 48.4 CAF with decorrelating signal operations (i. e., inserting a decorrelation unit or adding a probe signal $r[k]$)

From (48.16), we observe that the bias in the feedback path estimate $\hat{\boldsymbol{f}}[k]$ decreases with increasing power ratio of the feedback signal $n[k]$ (i. e., the signal to identify) to the desired signal $x[k]$ (which acts as a disturbance). Since

$$n[k] = F(q,k)C(q,k)x[k]\,, \tag{48.38}$$

the larger the loop gain $|G(q,k)F(q,k)|$, the smaller the bias will be. This indicates that the bias will be smallest for large gains $G(q,k)$ and for frequencies that are closest to instability, which is a desirable property.

48.2.3 Reducing the Bias of the CAF

Classical approaches to reduce the bias of the CAF include inserting signal decorrelation operations in the forward path $G(q,k)$ and reducing the adaptation speed of the feedback canceller.

Inserting Signal Decorrelation Operations

A first solution to reduce the bias is to include decorrelating signal operations in cascade with the forward path $G(q,k)$, as depicted in Fig. 48.4. Ideally, the decorrelation unit removes any correlation between the desired signal $x[k]$ and the input $u[k]$ to the adaptive filter $\hat{F}(q,k)$, so that the bias term in (48.32) equals zero. Often, the decorrelation unit however degrades the sound quality, making full decorrelation impossible.

For many signals $x[k]$, the autocorrelation sequence $\mathcal{E}\{x[k]x[k-\delta]\}$ decreases with increasing lag δ. A common technique to reduce the correlation is to increase the delay d_G by inserting a delay into the forward path [48.28]. Many audio signals however have a relatively strong autocorrelation. A large delay d_G may then be required to efficiently decorrelate $x[k]$ and $u[k]$. However, the delay d_G should not exceed 10 ms in order not to degrade intelligibility [48.29] and sound quality, e.g., by comb-filter effects resulting from destructive and constructive interference between the direct sound to the eardrum and the amplified and delayed sound [48.30, 31].

As an illustration, Fig. 48.5 depicts the misalignment of the PBFD CAF as a function of the delay d_G of $G(q,k) = Gq^{-d_G}$ with $G = 10\,\mathrm{dB}$ for the acoustic feedback path in Fig. 48.2 with a mobile phone attached to the head. Three desired signals $x[k]$ are considered: a white-noise signal, a stationary speech weighted noise signal, and a real speech signal. The real speech signal

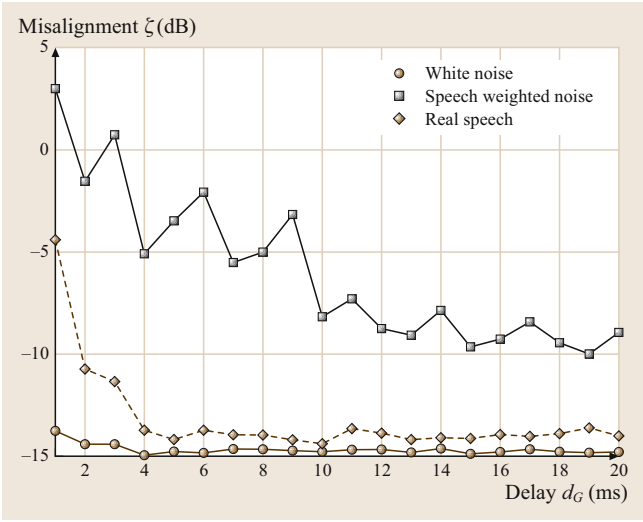


Fig. 48.5 Misalignment of the PBFD CAF as a function of the delay d_G for white noise, speech weighted noise, and real speech as a desired signal. $G(q,k) = Gq^{-d_G}$ with $G = 10\,\text{dB}$

consists of English sentences spoken by a male speaker from the hearing in noise test (HINT) database [48.32]. For speech weighted noise and real speech, the CAF is clearly biased. For both signals, the bias (or misalignment) decreases with increasing delay d_G. A delay of 4 ms provides sufficient decorrelation for the speech weighted noise signal. However, for the real speech signal, a much larger delay $d_G > 10\,\text{ms}$ is required.

Other techniques to reduce the correlation between $x[k]$ and $u[k]$ are inserting a *probe signal* $r[k]$ (noise signal) at the input $u[k]$ of the loudspeaker (see Fig. 48.4) or adding *nonlinearities* to the forward path $G(q,k)$, such as frequency shifting, phase or delay modulation [48.12, 33–35]. Such methods may, however, degrade the sound quality. In the former case, e.g., the probe signal should be inaudible so that it does not degrade the signal-to-noise ratio. Its level will thus have to be lower than the level of the ordinary output $u[k]$, resulting in a significant residual bias. A small amount of modulation or a small frequency shift (e.g., a frequency shift of up to 5 Hz [48.13]) may still be acceptable for speech signals, however, its effect on music is objectionable. These techniques will not be further discussed.

Reducing the Adaptation Speed

In [48.36–38], distortion of the desired signal $x[k]$ is reduced by keeping the adaptation speed of the feedback canceller small. When feedback or a sudden change in the feedback path is detected, the adaptation speed is temporarily increased. A slow adaptation speed reduces the ability to track changes in the spectrum of the desired signal $x[k]$, and hence, limits the negative effect of a temporary signal with a strong autocorrelation on the feedback path estimate [48.30]. Convergence of this technique may however be too slow to track variations in the feedback path successfully so that the maximum amplification may still be limited.

In addition to inserting signal decorrelating operations or reducing the adaptation speed, other approaches have been proposed in the literature for improving the estimation accuracy of the adaptive feedback canceller. A first class of techniques (described in Sect. 48.3) exploits prior knowledge of the acoustic feedback path to improve the adaptation of the feedback canceller. A second class of techniques (discussed in Sect. 48.4) views the feedback path as a part of a closed-loop system and applies closed-loop system identification theory.

48.3 Feedback Cancellation Based on Prior Knowledge of the Acoustic Feedback Path

In [48.15, 22, 39], prior knowledge of the acoustic feedback path is exploited to reduce the bias of the standard CAF. The adaptation of the feedback canceller is controlled based on the prior knowledge through constrained (Sect. 48.3.1) or bandlimited adaptation (Sect. 48.3.2).

48.3.1 Constrained Adaptation (C-CAF)

In [48.22, 39], the bias is reduced by constrained adaptation: the filter coefficients $\hat{\boldsymbol{f}}[k]$ of the adaptive feedback canceller $\hat{F}(q,k)$ are not allowed to deviate too much from reference filter coefficients $\hat{\boldsymbol{f}}_{\text{ref}}$. These reference filter coefficients $\hat{\boldsymbol{f}}_{\text{ref}}$ are measured during start-up or fitting, by means of a probe signal $r[k]$.

A cheap method to impose this constraint is to add a term to the cost function (48.12) of the feedback canceller that penalizes excessive deviation of the filter $\hat{\boldsymbol{f}}[k]$ from the reference filter $\hat{\boldsymbol{f}}_{\text{ref}}$, i. e.,

$$J(\hat{\boldsymbol{f}}[k]) = \mathcal{E}\left\{|y[k] - \hat{\boldsymbol{f}}^{\mathrm{T}}[k]\boldsymbol{u}[k]|^2\right\} + \eta(\hat{\boldsymbol{f}}[k] - \hat{\boldsymbol{f}}_{\text{ref}})^{\mathrm{T}}(\hat{\boldsymbol{f}}[k] - \hat{\boldsymbol{f}}_{\text{ref}})\,. \quad (48.39)$$

This results in the solution

$$\hat{\boldsymbol{f}}[k] = \left(\mathcal{E}\{\boldsymbol{u}[k]\boldsymbol{u}^{\mathrm{T}}[k]\} + \eta\mathbf{I}\right)^{-1} \left(\mathcal{E}\{\boldsymbol{u}[k]y[k]\} + \eta\hat{\boldsymbol{f}}_{\mathrm{ref}}\right) . \tag{48.40}$$

The scalar η trades off between signal distortion reduction and the ability to model deviations from $\hat{\boldsymbol{f}}_{\mathrm{ref}}$. The larger η, the more $\hat{\boldsymbol{f}}[k]$ evolves towards $\hat{\boldsymbol{f}}_{\mathrm{ref}}$.

Time-Domain NLMS

From (48.39), the following time-domain NLMS update equation can be derived

$$\hat{\boldsymbol{f}}[k+1] = \hat{\boldsymbol{f}}[k] + \frac{\bar{\mu}}{L_{\hat{F}}\left(\overline{|u[k]|^2} + \overline{|e[k]|^2}\right) + \eta + c} \times \left[\boldsymbol{u}[k]\left(y[k] - \hat{\boldsymbol{f}}^{\mathrm{T}}[k]\boldsymbol{u}[k]\right) - \eta\left(\hat{\boldsymbol{f}}[k] - \hat{\boldsymbol{f}}_{\mathrm{ref}}\right)\right] . \tag{48.41}$$

Partitioned-Block Frequency-Domain NLMS

Converting (48.41) into the partitioned-block frequency domain results in [48.27]:

$$\hat{\boldsymbol{F}}_p[\bar{k}+1] = \hat{\boldsymbol{F}}_p[\bar{k}] + \boldsymbol{\Delta}[\bar{k}]\left(\mathcal{F}\boldsymbol{g}\mathcal{F}^{-1}\boldsymbol{U}_p^{\mathrm{H}}[\bar{k}]\boldsymbol{E}[\bar{k}] - \boldsymbol{\eta}\left(\hat{\boldsymbol{F}}_p[\bar{k}] - \hat{\boldsymbol{F}}_{p,\,\mathrm{ref}}\right)\right) , \tag{48.42}$$

where

$$\hat{\boldsymbol{f}}_{p,\,\mathrm{ref}} = \left[\hat{f}_{\mathrm{ref},\,pP} \;\; \cdots \;\; \hat{f}_{\mathrm{ref},\,(p+1)P-1}\right]^{\mathrm{T}} , \tag{48.43}$$

$$\hat{\boldsymbol{F}}_{p,\,\mathrm{ref}} = \mathcal{F}\begin{bmatrix} \hat{\boldsymbol{f}}_{p,\,\mathrm{ref}} \\ \mathbf{0} \end{bmatrix} \begin{matrix} \updownarrow P \\ \updownarrow M-P \end{matrix} . \tag{48.44}$$

$\boldsymbol{\eta}$ is a diagonal matrix that contains the trade-off parameters η_m, $m = 0, \ldots, M-1$ for the different frequency bins m. The diagonal step-size matrix $\boldsymbol{\Delta}[\bar{k}]$ is

$$\boldsymbol{\Delta}[\bar{k}] = \mathrm{diag}\left\{\mu_0[\bar{k}], \;\; \ldots \;, \;\; \mu_{M-1}[\bar{k}]\right\} \tag{48.45}$$

with

$$\mu_m[\bar{k}] = \frac{\bar{\mu}_m}{\sum_{p=0}^{\frac{L_{\hat{F}}}{P}-1} |U_{p,m}[\bar{k}]|^2 + \frac{L_{\hat{F}}}{P}|E_{p,m}[\bar{k}]|^2 + \eta_m + c} . \tag{48.46}$$

In the sequel, the CAF with constrained adaptation will be referred to as C-CAF.

Performance

Figure 48.6 illustrates the effect of $\eta_m = \eta$ on the performance of the PBFD C-CAF for different delays d_G

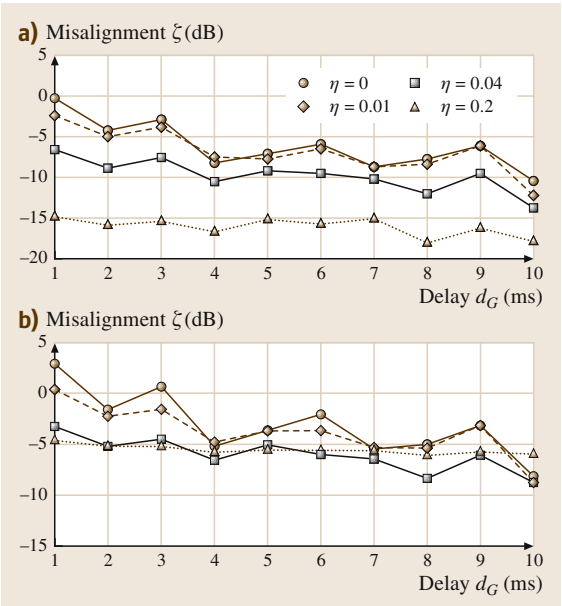


Fig. 48.6a,b Misalignment of the C-CAF as a function of the delay d_G of $G(q,k) = Gq^{-d_G}$ for different settings of η and real speech as a desired signal. **(a)** Without obstruction ($G = 17$ dB). **(b)** With a mobile phone attached to the head ($G = 10$ dB)

and real speech as a desired signal. The reference filter $\hat{\boldsymbol{f}}_{\mathrm{ref}}$ is a 20-taps filter. It was computed for the scenario without obstruction by disconnecting the forward path $G(q,k)$ and inserting a white-noise signal $r[k]$ into the loudspeaker. The frequency response of the reference filter $\hat{\boldsymbol{f}}_{\mathrm{ref}}$ together with the frequency response of the true feedback path is depicted in Fig. 48.7. The feedback path when a mobile phone is attached to the ear is also shown. Figure 48.6(a) shows the misalignment for the feedback path without obstruction. Figure 48.6b depicts the misalignment for the feedback path with a mobile phone attached to the head, illustrating the performance of the C-CAF in the presence of a deviation in the feedback path model. The forward path gain was set close to instability, i. e., $G = 17$ dB for the scenario without obstruction and $G = 10$ dB for the scenario with a mobile phone.

For the scenario without obstruction, the reference filter is a good model of the actual feedback path. As a result, the misalignment of the C-CAF decreases with increasing η. When the mobile phone is attached to the head, the actual feedback path deviates from the reference path (Fig. 48.6). If η is chosen too large, the feedback path deviation cannot be successfully mod-

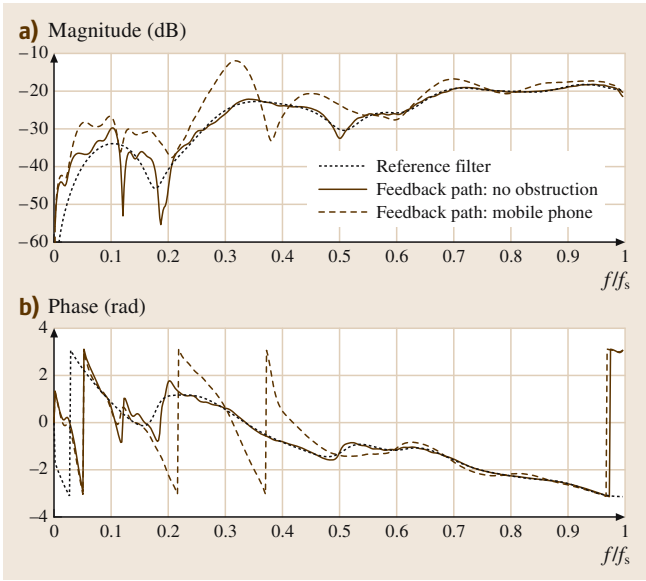


Fig. 48.7a,b Frequency response of the reference filter $\hat{\boldsymbol{f}}_{\text{ref}}$ used in the C-CAF. For comparison, the true feedback path for the scenario without obstruction and with a mobile phone are also depicted

eled. Indeed, Fig. 48.6b shows that for $\eta = 0.2$, a similar or worse misalignment is obtained than for $\eta = 0.04$. In the sequel, $\eta = 0.04$.

48.3.2 Bandlimited Adaptation (BL-CAF)

In [48.15, 38, 40–42], feedback cancellation is restricted to the frequency band that encompasses the unstable frequencies through bandlimited adaptation. Indeed, to avoid instability, it suffices that the feedback canceller cancels the feedback signal at the critical frequencies, where instability is about to occur. Typically, the acoustic feedback path of a hearing aid provides less attenuation at high frequencies, where most hearing aid users have the largest hearing loss (Fig. 48.2) [48.3]. As a result, the risk for instability is often highest in the high-frequency range, while most of the desired signal energy is typically concentrated at low frequencies, e.g., for speech signals. By concentrating the adaptation of the feedback canceller on the higher frequencies only, the feedback canceller may be more efficient and will introduce less distortion.

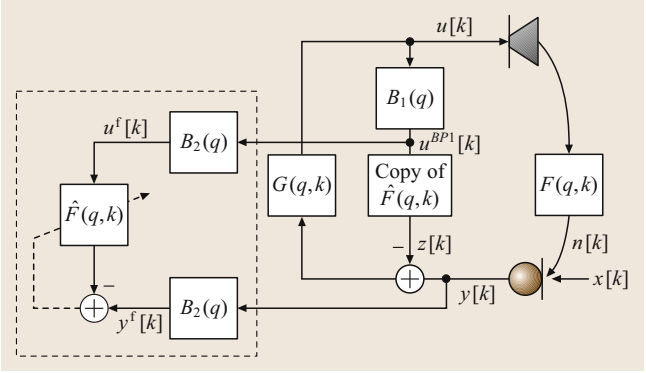


Fig. 48.8 Bandlimited adaptive feedback cancellation (BL-CAF)

Figure 48.8 depicts the block diagram of the bandlimited feedback canceller (abbreviated as BL-CAF). The filters $B_1(q)$ and $B_2(q)$ are high- or bandpass filters chosen such that all critical frequencies are preserved while the desired signal components are removed as much as possible. The filter $B_1(q)$ in the feedback cancellation path limits the feedback cancellation signal to the frequency band of interest and hence prevents the desired signal components outside the critical band from being distorted by the feedback canceller $\hat{F}(q,k)$. To focus the modeling effort of the adaptive feedback canceller $\hat{\boldsymbol{f}}[k]$ on the regions that contain the critical frequencies, the adaptive filter minimizes the frequency-weighted error energy

$$J(\hat{\boldsymbol{f}}[k]) = \mathcal{E}\left\{|B_2(q)(y[k] - \hat{F}(q,k)u^{\text{BP}_1}[k])|^2\right\} = \mathcal{E}\left\{|y^{\text{f}}[k] - \hat{F}(q,k)u^{\text{f}}[k]|^2\right\}, \quad (48.47)$$

where

$$u^{\text{BP}_1}[k] = B_1(q)u[k], \quad (48.48)$$

$$y^{\text{f}}[k] = B_2(q)y[k], \quad (48.49)$$

$$u^{\text{f}}[k] = B_2(q)u^{\text{BP}_1}[k]\,. \quad (48.50)$$

The filter $B_2(q)$ reduces the energy of the residual error signal (which acts as a disturbance to the adaptive feedback canceller). As a result, the misadjustment and bias of the bandlimited adaptive feedback canceller is reduced compared to the standard CAF.

Note that the filter $B_1(q)$ introduces a group delay in the feedback cancellation path. This group delay should not exceed the delay in the acoustic feedback path, otherwise the feedback signal cannot be cancelled [48.15]. To guarantee a small group delay and to save computational complexity, low-order elliptic IIR filters $B_1(q)$ and $B_2(q)$ are used in [48.15].

In [48.15], minimization of (48.47) is realized through a filtered-X algorithm. In the filtered-X algorithm, the input signal $u^{\text{BP}_1}[k]$ and the feedback-compensated signal $e[k] = y[k] - \hat{F}(q,k)u^{\text{BP}_1}[k]$ are

filtered by $B_2(q)$ before being used to update the adaptive filter $\hat{F}(q,k)$. Equivalently, $J(\boldsymbol{f}[\bar{k}])$ can be minimized by performing standard adaptive filtering techniques on the prefiltered microphone and loudspeaker data $y^{\mathrm{f}}[k]$ and $u^{\mathrm{f}}[k]$ (as illustrated in Fig. 48.8) [48.27]. In contrast to the filtered-X algorithm, any change to the adaptive filter $\hat{\boldsymbol{f}}[k]$ then has an immediate effect on the frequency-weighted adaptation error even if $B_2(q)$ has a group delay. As a result, no stability problems may occur [48.15, 43].

Time-Domain NLMS

For NLMS adaptation, the update equation becomes

$$\hat{\boldsymbol{f}}[k] = \hat{\boldsymbol{f}}[k-1] + \mu[k]\boldsymbol{u}[k]\underbrace{\left(y^{\mathrm{f}}[k] - \hat{\boldsymbol{f}}^{\mathrm{T}}[k-1]\boldsymbol{u}^{\mathrm{f}}[k]\right)}_{e^{\mathrm{f}}[k]}, \tag{48.51}$$

with

$$\boldsymbol{u}^{\mathrm{f}}[k] = \left[u^{\mathrm{f}}[k] \;\; \ldots \;\; u^{\mathrm{f}}[k-L_{\hat{F}}]\right] \tag{48.52}$$

$$\mu[k] = \frac{\bar{\mu}}{L_{\hat{F}}\left(\overline{|u^{\mathrm{f}}[k]|^2} + \overline{|e^{\mathrm{f}}[k]|^2}\right) + c}, \tag{48.53}$$

Partitioned-Block Frequency-Domain NLMS

The PBFD implementation equals [48.27]

$$\hat{\boldsymbol{F}}_p[\bar{k}+1] = \hat{\boldsymbol{F}}_p[\bar{k}] + \boldsymbol{\Delta}[\bar{k}]\mathcal{F}\boldsymbol{g}\mathcal{F}^{-1}\boldsymbol{U}_p^{\mathrm{f,H}}[\bar{k}]\boldsymbol{E}^{\mathrm{f}}[\bar{k}], \tag{48.54}$$

where

$$\boldsymbol{U}_p^{\mathrm{f}}[\bar{k}] = \mathrm{diag}\left\{\mathcal{F}\begin{bmatrix} u^{\mathrm{f}}[(\bar{k}+1)L - pP - M + 1] \\ \vdots \\ u^{\mathrm{f}}[(\bar{k}+1)L - pP] \end{bmatrix}\right\}, \tag{48.55}$$

$$\boldsymbol{E}^{\mathrm{f}}[\bar{k}] = \mathcal{F}\begin{bmatrix}\boldsymbol{0} \\ \mathbf{I}_L\end{bmatrix}\left(\boldsymbol{y}_{\bar{k}}^{\mathrm{f}} - \begin{bmatrix}\boldsymbol{0} & \mathbf{I}_L\end{bmatrix}\mathcal{F}^{-1} \times \sum_{p=0}^{\frac{L_{\hat{F}}}{P}-1} \boldsymbol{U}_p^{\mathrm{f}}[\bar{k}]\hat{\boldsymbol{F}}_p[\bar{k}]\right), \tag{48.56}$$

$$\boldsymbol{y}_{\bar{k}}^{\mathrm{f}} = \left[y[\bar{k}L+1] \;\; \ldots \;\; y[(\bar{k}+1)L]\right]^{\mathrm{T}}. \tag{48.57}$$

The step sizes $\mu_m[\bar{k}]$ are computed as

$$\frac{\bar{\mu}_m}{\sum_{p=0}^{\frac{L_{\hat{F}}}{P}-1}\left|U_{p,m}^{\mathrm{f}}[\bar{k}]\right|^2 + \frac{L_{\hat{F}}}{P}\left|E_{p,m}^{\mathrm{f}}[\bar{k}]\right|^2 + c}. \tag{48.58}$$

Performance

The bandwidths and the stopband attenuation of the filters $B_1(q)$ and $B_2(q)$ determine the efficiency of the bandlimited feedback canceller $\hat{F}(q,k)$: the larger the bandwidth (or the smaller the stopband attenuation), the larger the frequency range where acoustic feedback is cancelled, but the larger the bias and misadjustment of the feedback path estimate. To specify the bandwidth of $B_1(q)$ and $B_2(q)$, identification of the critical frequency region for different hearing aids and different hearing aid users is required, e.g., during fitting. However, the critical frequency region expands with increasing gain in the forward path $G(q,k)$ as well as, e.g., with the presence of a telephone handset or hand palm close to the ear (Fig. 48.2). Hence, to guarantee stability, the bandwidth and stopband attenuation of $B_1(q)$ and

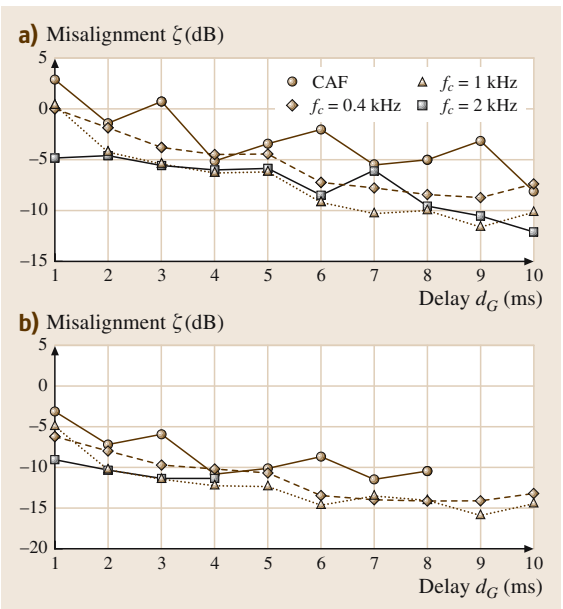


Fig. 48.9a,b Misalignment of BL-CAF as a function of the delay d_G for different cut-off frequencies f_c (i. e., f_c = 0.4 kHz, f_c = 1 kHz, and f_c = 2 kHz) and real speech as a desired signal. **(a)** G = 10 dB. **(b)** G = 17 dB (for d_G = 5–6 ms and d_G = 8–10 ms, instability occurs for the BL-CAF with f_c = 2 kHz; for d_G = 9–10 ms, instability occurs for the CAF)

$B_2(q)$ should not be chosen too small. In this Chapter, a second-order IIR filter $B_1(q) = B_2(q)$ with a cut-off frequency of 1 kHz and a stopband attenuation of 20 dB is used.

As an illustration, Fig. 48.9 depicts the misalignment of the BL-CAF as a function of the forward path delay d_G for three different cut-off frequencies f_c, i. e., $f_c =$ 0.4 kHz, $f_c = 1$ kHz and $f_c = 2$ kHz. For comparison, the misalignment of the CAF is also shown. The acoustic feedback path with a mobile phone attached to the head is used. The desired signal $x[k]$ is a real speech signal. Two forward path gains G are depicted, i.e., $G = 10$ dB and $G = 17$ dB. For all delays d_G, the BL-CAF achieves a smaller misalignment than the CAF, especially for low gains G. For $G = 10$ dB, a cut-off frequency of 1 kHz or 2 kHz results in a smaller misalignment than a cut-off frequency of 0.5 kHz. For $G = 17$ dB, the cut-off frequency of 2 kHz performs worse than $f_c = 1$ kHz: for $d_G \geq 5$ ms, instability even occurs. At high gains, the region with critical frequencies increases, explaining the worse performance. This illustrates that f_c should not be chosen too large.

48.4 Feedback Cancellation Based on Closed-Loop System Identification

All of the aforementioned feedback cancellation techniques initially ignore the presence of the closed signal loop $G(q, k)$ and apply standard adaptive filtering techniques for open-loop systems. In a second stage, signal processing operations are included to reduce the negative effect of $G(q, k)$ on the performance.

An alternative approach is to view the feedback path immediately as a part of a closed-loop system and to apply closed-loop system identification theory [48.1, 17, 30, 44, 45]. Closed-loop identification methods can be classified as direct, indirect, or joint input–output methods [48.1]. In contrast to the indirect and the joint input–output approach, the direct method requires neither assumptions about the forward path $G(q, k)$ (e.g., linearity) nor the presence of an external probe signal $r[k]$, making it an appealing approach for feedback cancellation.

Section 48.4.1 defines the closed-loop system setup. Section 48.4.2 describes the direct method of closed-loop system identification in the context of feedback cancellation. In this method, the bias of the standard CAF is reduced by incorporating a (stationary or time-varying) model of the desired signal $x[k]$ in the identification [48.18, 30, 45, 46]. The estimation of the desired signal model is discussed in Sect. 48.4.3. Section 48.4.4 discusses the other two closed-loop system identification procedures, i. e., the indirect method and the joint input–output method [48.17, 47].

48.4.1 Closed-Loop System Setup

Figure 48.10 depicts the closed-loop system setup. The *open-loop system to be identified* is given by

$$y[k] = F(q, k)u[k] + x[k] \,. \tag{48.59}$$

For the time being, we assume that the desired signal $x[k]$ can be modeled as

$$x[k] = H(q, k)w[k] \,, \tag{48.60}$$

with $w[k]$ a zero-mean white-noise sequence and $H(q, k)$ monic and inversely stable.

The output signal $y[k]$ is fed back to the input $u[k]$ according to

$$u[k] = G(q, k)(y[k] - \hat{F}(q, k)u[k]) + r[k] \,. \tag{48.61}$$

The signal $r[k]$ is an optional probe signal that is independent of $x[k]$, and can be viewed as an extra excitation signal of the feedback path $F(q, k)$. The probe signal is generally a noise signal.

By combining (48.59) and (48.61), we obtain the closed-loop relations [48.1, 44]

$$y[k] = (1 + F(q, k)C(q, k))x[k] + F(q, k)S(q, k)r[k] \,, \tag{48.62}$$

$$u[k] = C(q, k)x[k] + S(q, k)r[k] \,, \tag{48.63}$$

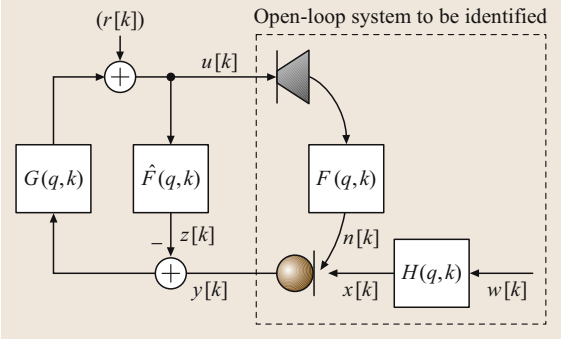


Fig. 48.10 Closed-loop system set-up

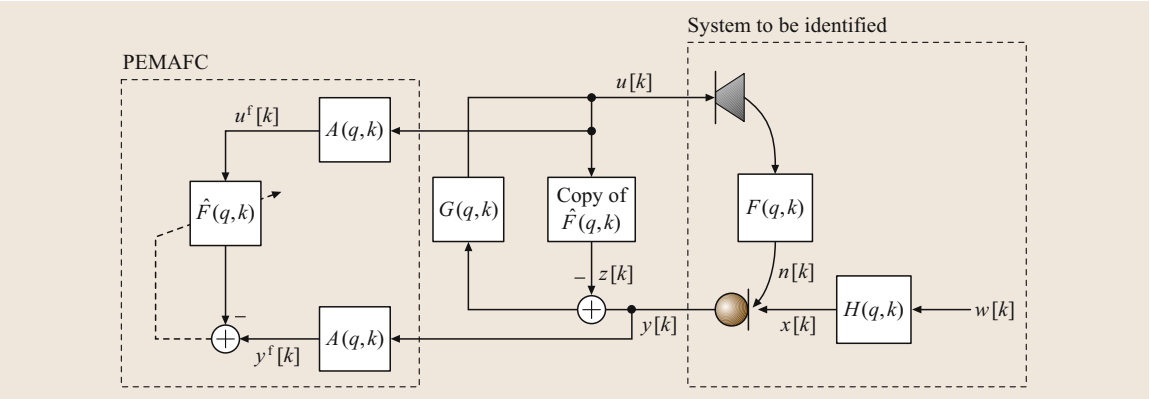


Fig. 48.11 Feedback cancellation with the prediction error method (PEMAFC)

where $S(q,k)$ is the so-called sensitivity function

$$S(q,k) = \frac{1}{1 - G(q,k)(F(q,k) - \hat{F}(q,k))} \tag{48.64}$$

and $C(q,k)$ is the closed-loop transfer function from $x[k]$ to $u[k]$ [see(48.9)]

$$C(q,k) = S(q,k)G(q,k)\,. \tag{48.65}$$

For future use, we denote the closed-loop transfer function from $r[k]$ to $y[k]$ as $F^c(q,k)$:

$$F^c(q,k) = F(q,k)S(q,k)\,. \tag{48.66}$$

Depending on the assumptions that are made about the forward path $G(q,k)$, closed-loop identification methods can be classified as *indirect, joint input–output*, or *direct* methods [48.1, 44].

48.4.2 Direct Method

In the direct method of closed-loop identification, the open-loop system $\{F(q,k),\ H(q,k)\}$

$$y[k] = F(q,k)u[k] + H(q,k)w[k]\,, \tag{48.67}$$

is identified from the loudspeaker signal $u[k]$ and the microphone signal $y[k]$ by an open-loop identification method, thereby ignoring the presence of the closed signal loop $G(q,k)$ [48.18, 30, 45, 48]. Only some specific open-loop identification methods can be applied to closed-loop systems, such as the prediction error method (PEM) [48.1, 44]. The PEM produces an estimate $\hat{F}(q,k)$ and $\hat{H}(q,k)$ of the feedback path $F(q,k)$ and the desired signal model $H(q,k)$, respectively, by minimizing the energy of the so-called prediction error

$$e^p[k] = \hat{H}(q,k)^{-1}(y[k] - \hat{F}(q,k)u[k])\,,$$

i. e.,

$$\begin{aligned} J(\hat{\boldsymbol{f}}[k]) &= \mathcal{E}\{|\hat{H}(q,k)^{-1}(y[k] - \hat{F}(q,k)u[k])|^2\}\,, \\ &= \mathcal{E}\{|y^f[k] - \hat{\boldsymbol{f}}^T[k]\boldsymbol{u}^f[k]|^2\}\,, \end{aligned} \tag{48.68}$$

where

$$\boldsymbol{u}^f[k] = \left[u^f[k] \ \ldots \ u^f[k - L_{\hat{F}} + 1]\right]^T\,, \tag{48.69}$$

$$y^f[k] = \hat{H}(q,k)^{-1}y[k]\,, \tag{48.70}$$

$$u^f[k] = \hat{H}(q,k)^{-1}u[k]\,. \tag{48.71}$$

Minimization of (48.68) results in

$$\hat{\boldsymbol{f}}[k] = \mathcal{E}\{\boldsymbol{u}^f[k]\boldsymbol{u}^{f,\,T}[k]\}^{-1}\mathcal{E}\{\boldsymbol{u}^f[k]y^f[k]\}\,, \tag{48.72}$$

where $y^f[k]$ can be decomposed as:

$$y^f[k] = \hat{H}(q,k)^{-1}x[k] + F(q,k)u^f[k]\,. \tag{48.73}$$

If $\hat{H}(q,k) = H(q,k)$,

$$y^f[k] = w[k] + F(q,k)u^f[k]\,. \tag{48.74}$$

The desired signal $x[k]$ is converted into a white-noise signal $w[k]$. As a result, $\hat{\boldsymbol{f}}[k]$ results in an unbiased feedback path estimate in the sufficient-order case (i. e., $L_{\hat{F}} = L_F$) [48.18, 45].

From (48.68) it follows that $J(\hat{\boldsymbol{f}}[k])$ can be minimized by performing standard adaptive filtering techniques on the pre-whitened data $y^f[k]$ and $u^f[k]$ [48.43]. This is illustrated in Fig. 48.11, where $A(q,k) = \hat{H}(q,k)^{-1}$. The update equations of the time-domain and PBFD NLMS algorithm correspond to (48.51–48.58) with $y^f[k]$ and $u^f[k]$ now defined in (48.70–48.71) (Table 48.2). In the sequel, we refer to the feedback canceller based on the PEM as PEMAFC.

Table 48.2 PBFD implementation of PEMAFC-a

For each block of L input samples $[u[\bar{k}L+1], \ldots, u[(\bar{k}+1)L]$

Block of output samples $z_{\bar{k}} = [z[\bar{k}L+1] \ \ldots \ z[(\bar{k}+1)L]]^{\mathrm{T}}$

$$z_{\bar{k}} = [\boldsymbol{0} \ \boldsymbol{I}_L]\mathcal{F}^{-1}\sum_{p=0}^{\frac{L_{\hat{F}}}{P}-1}\boldsymbol{U}_p[\bar{k}]\hat{\boldsymbol{F}}_p[\bar{k}]\,,$$

with

$$\boldsymbol{U}_p[\bar{k}] = \mathrm{diag}\left\{\mathcal{F}\begin{bmatrix} u[(\bar{k}+1)L-pP-M+1] \\ \vdots \\ u[(\bar{k}+1)L-pP]\end{bmatrix}\right\},\ p=0,\ \ldots,\ \frac{L_{\hat{F}}}{P}-1\,.$$

Prefilter loudspeaker and microphone signal with $A(q,\bar{k}_a-1)$, $\bar{k}_a = \left\lceil\frac{\bar{k}L+i}{N}\right\rceil$ (N is the framelength):

$$u^{\mathrm{f}}[\bar{k}L+i] = A(q,\bar{k}_a-1)u[\bar{k}L+i-N]\,,$$
$$y^{\mathrm{f}}[\bar{k}L+i] = A(q,\bar{k}_a-1)y[\bar{k}L+i-N],\ i=1,\ \ldots,\ L,$$
$$\boldsymbol{U}_p^{\mathrm{f}}[\bar{k}] = \mathrm{diag}\left\{\mathcal{F}\begin{bmatrix} u^{\mathrm{f}}[(\bar{k}+1)L-pP-M+1] \\ \vdots \\ u^{\mathrm{f}}[(\bar{k}+1)L-pP]\end{bmatrix}\right\}.$$

Update AR model $A(q,\bar{k}_a)$:

If $\bar{k}L+i = \bar{k}_a N$ with $\bar{k}_a$ an integer:

$$A(q,\bar{k}_a) = \text{Levinson–Durbin}\left([e(\bar{k}L+i-N+1] \ \ldots \ e[\bar{k}L+i]]^{\mathrm{T}}\right)$$

with $e[\bar{k}L+i] = y[\bar{k}L+i] - z[\bar{k}L+i]$.

Update formula:

$$\boldsymbol{y}_{\bar{k}}^{\mathrm{f}} = [y^{\mathrm{f}}[\bar{k}L+1] \ \ldots \ y^{\mathrm{f}}[(\bar{k}+1)L]]^{\mathrm{T}}\,,$$
$$\boldsymbol{E}^{\mathrm{f}}[\bar{k}] = \mathcal{F}\begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{I}_L\end{bmatrix}\left(\boldsymbol{y}_{\bar{k}}^{\mathrm{f}} - [\boldsymbol{0} \ \boldsymbol{I}_L]\mathcal{F}^{-1}\sum_{p=0}^{\frac{L_{\hat{F}}}{P}-1}\boldsymbol{U}_p^{\mathrm{f}}[\bar{k}]\hat{\boldsymbol{F}}_p[\bar{k}]\right),$$
$$\hat{\boldsymbol{F}}_p[\bar{k}+1] = \hat{\boldsymbol{F}}_p[\bar{k}] + \boldsymbol{\Delta}[\bar{k}]\mathcal{F}\boldsymbol{g}\mathcal{F}^{-1}\boldsymbol{U}_p^{\mathrm{f},\mathrm{H}}[\bar{k}]\boldsymbol{E}^{\mathrm{f}}[\bar{k}]\,.$$

Step size:

$$\boldsymbol{\Delta}[\bar{k}] = \mathrm{diag}\{\mu_0[\bar{k}],\ \ldots,\ \mu_{M-1}[\bar{k}]\}\,,$$
$$\mu_m[\bar{k}] = \frac{\bar{\mu}_m}{\sum_{p=0}^{\frac{L_{\hat{F}}}{P}-1}\left|U_{p,m}^{\mathrm{f}}[\bar{k}]\right|^2 + \frac{L_{\hat{F}}}{P}\left|E_{p,m}^{\mathrm{f}}[\bar{k}]\right|^2 + c}\,.$$

48.4.3 Desired Signal Model

The PEMAFC requires an estimate of the desired signal model $H(q,k)$. Both, fixed and adaptive estimates of the desired signal model $H^{-1}(q,k)$ have been considered in the literature. In the sequel, PEMAFC with a fixed model is referred to as PEMAFC-f and PEMAFC with an adaptive model is referred to as PEMAFC-a.

Fixed Desired Signal Model (PEMAFC-f)

In [48.30, 45], the desired signal model $H(q,k)$ is approximated by a *fixed* low-pass all-pole filter $\hat{H}(q,k) = A^{-1}(q)$, which represents the long-term average speech spectrum. A simple model for the average speech spectrum is a first-order all-pole model:

$$\hat{H}(q,k) = \frac{1}{1-\alpha q^{-1}}\,, \tag{48.75}$$

with $\alpha < 1$ (e.g., $\alpha = 0.9$) [48.22].

Adaptive Desired Signal Model (PEMAFC-a)

In practice, the desired signal model $H^{-1}(q,k)$ is unknown and highly time varying. In addition, the quality of the feedback canceller $\hat{F}(q,k)$ strongly depends on the accuracy of the signal model estimate $\hat{H}(q,k)$ [48.45], so that it is desirable to identify the feedback path $F(q,k)$ as well as the desired signal model $H(q,k)$ with the PEM method. However, $F(q,k)$ and $H(q,k)$ are not always identifiable in the closed-loop system at hand [48.45].

Since most audio signals $x[k]$ can be approximated by a low-order AR model, it is assumed that

$$H^{-1}(q,k) = A(q,k) = 1 + q^{-1}\bar{A}(q,k) \tag{48.76}$$

with $\bar{A}(q,k)$ an FIR filter. In [48.18], it has been shown that the AR model $A(q,k)$ and the feedback path $F(q,k)$ can be both identified in closed-loop without adding nonlinearities or a probe signal, if the delay $d_G \geq L_{H^{-1}} = L_A$.

Since the filter length L_A is short, the required processing delay d_G is usually much shorter (e.g., 1–2 ms) than the delay d_G that is needed in the standard CAF to decorrelate $x[k]$ and $u[k]$. For example, a 10–20 ms speech segment at a sampling frequency $f_s = 16$ kHz can be modeled by a 10–20-order AR model with $w[k]$ a white-noise excitation (in case of unvoiced sounds) or a pulse-train excitation (in case of voiced sounds).

The AR model $A(q,k)$ is computed through linear prediction of the feedback compensated signal

$$e[k] = y[k] - \hat{\boldsymbol{f}}^{\mathrm{T}}[k]\boldsymbol{u}[k]\,, \tag{48.77}$$

on subsequent frames of $N = 160$ samples (i. e., 10 ms), e.g., using the Levinson–Durbin algorithm [48.49]. Note that $e[k] = x[k]$, if $\hat{\boldsymbol{f}}[k] = \boldsymbol{f}[k]$.

So far, we have assumed that the excitation $w[k]$ of $H(q,k)$ is a white-noise sequence. This assumption does not apply for voiced speech, where the excitation $w[k]$ approximates a pulse train that is periodical with the pitch period P (expressed as number of samples). A low-order (e.g., 10th to 20th order) AR model will not suffice to notch out the discrete frequencies caused by the pulse train excitation. As a result, the prefiltered loudspeaker signal $u^{\mathrm{f}}[k]$ will still be correlated with the excitation signal $w[k]$ at the pitch frequency $\frac{f_s}{P}$ and its harmonics $k\frac{f_s}{P}$ with $k \in \mathbb{N}$ and $\frac{k}{P} < \frac{1}{2}$. The residual correlation caused by the pulse train excitation can be removed by cascading the low-order short-term AR model $A_{\mathrm{ST}}(q,k)$ with a long-term predictor $A_{\mathrm{LT}}(q,k) = (1 - bq^{-\hat{P}[k]})$, where $\hat{P}[k]$ is an estimate of the pitch period P [48.48]:

$$A(q,k) = A_{\mathrm{ST}}(q,k)(1 - bq^{-\hat{P}[k]})\,. \tag{48.78}$$

For speech, the pitch frequency f_s/P lies between 50 and 400 Hz and hence for $f_s = 16$ kHz, P lies between 40 and 320 samples. To guarantee identifiability, the total filter length $\hat{P}[k] + L_{A_{\mathrm{ST}}}$ of $A(q,k)$ should not exceed the maximum allowable processing delay d_G.

For feedback cancellers $\hat{F}(q,k)$ with a short filter length $L_{\hat{F}} \leq P$ and for small delays d_G, the bias in the feedback path estimate due to the periodic pulse train excitation $w[k]$ is negligible [48.27]. In hearing aids, the dominant part of the feedback path $F(q,k)$ is quite short, so that a relatively short filter length $L_{\hat{F}}$ is typically used, i. e., $L_{\hat{F}} \leq 100$ [48.14]. In addition, the delay d_G should be kept small (i. e., ≤ 10 ms). As a result, the benefit of a long-term predictor is limited for hearing-aid applications.

The equations of the PBFD NLMS implementation of the PEMAFC-a are summarized in Table 48.2.

Performance

As an illustration, Fig. 48.12 depicts the misalignment of PEMAFC-f (with the fixed AR model (48.75)) and PEMAFC-a (with $L_A = 21$) as a function of the delay d_G for the acoustic feedback path with a mobile phone attached to the head and a forward path gain $G = 10$ dB. (To guarantee identifiability, $L_A = 16$ for $d_G = 1$ ms.) For comparison, the misalignment of the CAF is also shown.

In Fig. 48.12a, the desired signal $x[k]$ is speech weighted noise, generated by passing white noise through a 20th-order all-pole model. In this example, the desired signal $x[k]$ can be perfectly whitened by a 20th-order AR model $A(q,k)$, since the white-noise assumption is satisfied and $L_{H^{-1}} = 21$. Indeed, for $d_G \leq 2$ ms, the PEMAFC succeeds to fully decorrelate $x[k]$ and $u[k]$: the same misalignment is obtained as for the CAF with white noise as a desired signal $x[k]$ (Fig. 48.5). (For $d_G = 16$ ms, the AR model is undermodeled since $L_A = 16$, explaining the worse performance.) The fixed all-pole model in the PEMAFC-f whitens the desired signal to some extent such that better performance than the

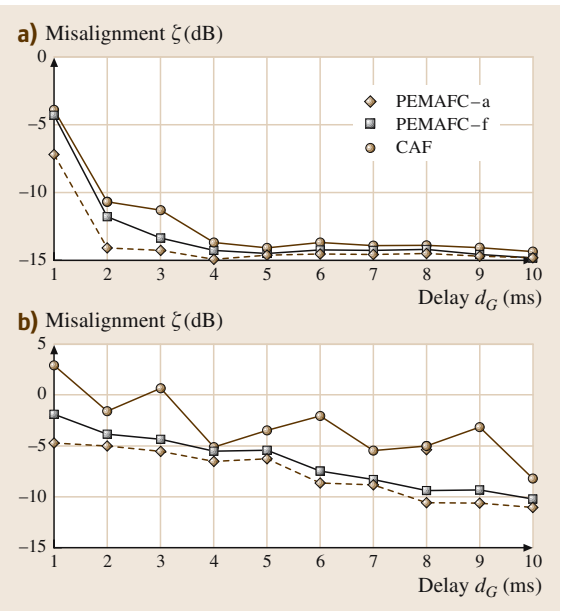


Fig. 48.12a,b Misalignment of the PEMAFC with a fixed (PEMAFC-f) and adaptive AR model (PEMAFC-a) as a function of the delay d_G for the acoustic feedback path with a mobile phone attached to the head. **(a)** $x[k]$ = speech weighted noise. **(b)** $x[k]$ = real speech

CAF is achieved. For $d_G \geq 4$ ms, $x[k]$ and $u[k]$ are uncorrelated so that all methods achieve similar performance.

In Fig. 48.12b, $x[k]$ is a real speech signal (male sentences from the HINT database [48.32]). The model assumptions for $x[k]$ (i. e., the assumption of a AR-model for $H^{-1}(q,k)$ and white noise for $w[k]$) are not completely satisfied, explaining the residual bias. However, for all delays $d_G \leq 10$ ms, the PEMAFC-f and PEMAFC-a achieve better performance than the CAF. PEMAFC-a outperforms PEMAFC-f thanks to the adaptive AR model.

48.4.4 Indirect and Joint Input–Output Method

In contrast to the direct method, the indirect and joint input–output method both require the use of a probe signal $r[k]$.

Indirect Method

In addition to the use of a probe signal $r[k]$, the indirect method [48.1, 44] requires knowledge of the forward path $G(q,k)$. This makes it not very suitable for hearing aids because $G(q,k)$ often contains user-dependent and adaptive signal processing features, e.g., dynamic-range compression.

In the indirect method, the closed-loop transfer function $F^{\mathrm{c}}(q,k)$ (see (48.66)) from $r[k]$ to $y[k]$ is estimated by means of an adaptive filter $\hat{F}^{\mathrm{c}}(q,k)$. If the forward path $G(q,k)$ is known, the feedback path estimate $\hat{F}(q,k)$ can then be extracted from this closed-loop transfer function estimate $\hat{F}^{\mathrm{c}}(q,k)$. In [48.50, 51], the estimate of the closed-loop transfer function $F^{\mathrm{c}}(q,k)$ is used as an approximation of $F(q,k)$, i. e., the sensitivity function $S(q,k)$ is assumed to be 1. This generally results in a biased feedback path estimate $\hat{F}(q,k)$, especially close to or at instability. In [48.52], it is shown that the first L_F coefficients of $F^{\mathrm{c}}(q,k)$ are equal to the coefficients of $F(q,k)$ if the forward path delay $d_G \geq L_{\hat{F}}$. For a white-noise signal $r[k]$ and $L_{\hat{F}} = L_F$, the estimate $\hat{F}^{\mathrm{c}}(q,k)$ is then unbiased.

The identification of the closed-loop system $F^{\mathrm{c}}(q,k)$ is seriously degraded by the presence of the desired signal $x[k]$ (see (48.62)). As a result, a large noise level or a slow adaptation speed is required to obtain an accurate estimate $\hat{F}^{\mathrm{c}}(q,k)$ (i. e., with a small variance).

Joint Input–Output Method

In [48.47], the joint input–output method is applied to obtain an unbiased feedback path estimate $\hat{F}(q,k)$. In this approach, the forward path $G(q,k)$ is assumed to have a certain structure (e.g., a linear filter) and the microphone signal $y[k]$ and the loudspeaker signal $u[k]$ are jointly viewed as the output of a system driven by the probe signal $r[k]$ (see (48.62)-(48.63)) [48.1, 44]:

$$\begin{bmatrix} y[k] \\ u[k] \end{bmatrix} = \begin{bmatrix} F^{\mathrm{c}}(q,k) \\ S(q,k) \end{bmatrix} r[k] + \begin{bmatrix} 1 + F(q,k)C(q,k) \\ C(q,k) \end{bmatrix} x[k] \,. \tag{48.79}$$

The closed-loop transfer function $F^{\mathrm{c}}(q,k)$ and the sensitivity function $S(q,k)$ are determined by means of standard adaptive filtering techniques performed on the excitation signal $r[k]$. The feedback path estimate $\hat{F}(q,k)$ is then derived from these two model estimates $\hat{F}^{\mathrm{c}}(q,k)$ and $\hat{S}(q,k)$ as:

$$\hat{F}(q,k) = \hat{F}^{\mathrm{c}}(q,k)\hat{S}^{-1}(q,k) \,. \tag{48.80}$$

In [48.47], $\hat{F}(q,k)$ is realized as an adaptive FIR filter that minimizes the error energy

$$\mathcal{E}\{|\hat{F}^{\mathrm{c}}(q,k)r[k] - \hat{F}(q,k)\hat{S}(q,k)r[k]|^2\} \,. \tag{48.81}$$

In [48.17], the feedback path $F(q,k)$ is identified with a two-stage method, requiring only two adaptive filters. In a first stage, the sensitivity function $S(q,k)$ from $r[k]$ to $u[k]$ is determined using a standard adaptive filter. The output $z_1[k] = \hat{S}(q,k)r[k]$ of the first adaptive filter is then fed to the input of a second adaptive filter $\hat{F}(q,k)$ that minimizes

$$\mathcal{E}\left\{\left|y[k] - \hat{F}(q)z_1[k]\right|^2\right\} \,. \tag{48.82}$$

Both implementations require the use of an external identification signal $r[k]$. The identification of $F^{\mathrm{c}}(q,k)$, $S(q,k)$ and $F(q,k)$ is impeded by the presence of the desired signal $x[k]$. To preserve sound quality, the level of the probe signal $r[k]$ should be low with respect to the desired output $G(q)x[k]$. To avoid a large variance in the feedback path estimate, a slow adaptation speed should be used for the adaptive filters [48.47], at the expense of a poor tracking performance. In addition, large filter lengths are required to identify the closed-loop transfer functions $\hat{F}^{\mathrm{c}}(q,k)$ and $\hat{S}(q,k)$ accurately, as they can easily contain narrowband peaks when the closed-loop system $F^{\mathrm{c}}(q,k)$ is close to instability (e.g., for high forward path gains).

48.5 Comparison

This section compares the performance of different adaptive feedback cancellation techniques. We concentrate on techniques that do not require the use of an external probe signal $r[k]$ during adaptation, i. e., CAF, C-CAF, BL-CAF, and PEMAFC (with fixed and adaptive AR model).

48.5.1 Steady-State Performance

Figure 48.13 compares the steady-state performance, i. e., the misalignment upon convergence, of the C-CAF ($\eta = 0.04$), BL-CAF, PEMAFC-f, and PEMAFC-a for the acoustic feedback path with a mobile phone attached to the head and real speech as a desired signal $x[k]$. The forward path gain is 10 dB. Figure 48.13a depicts the misalignment of CAF, C-CAF, and BL-CAF as a function of the delay d_G; Fig. 48.13b compares the misalignment of CAF, PEMAFC-f, and PEMAFC-a. For all delays $d_G \in [1, 10]$ ms, C-CAF, BL-CAF, and PEMAFC achieve a smaller misalignment than the CAF: depending on the delay d_G and the algorithm, a reduction in misalignment of 1 to 8.5 dB is achieved. PEMAFC-a generally achieves the best performance. PEMAFC-f performs 1–2 dB worse than the PEMAFC-a. For delays $d_G > 3$ ms, BL-CAF and PEMAFC-a have about the same steady-state misalignment.

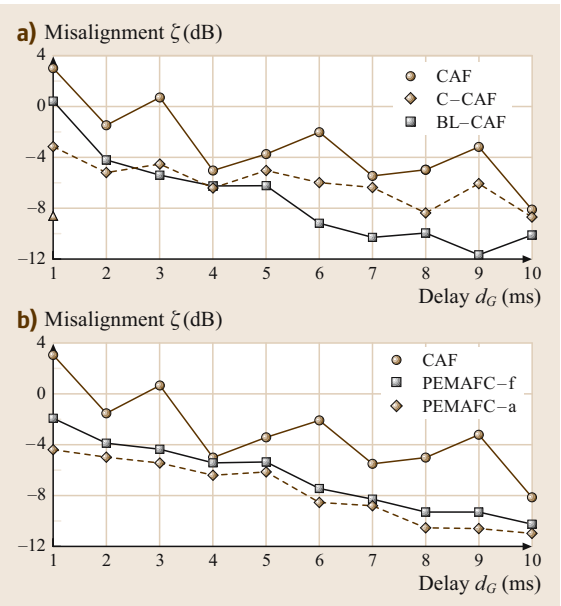


Fig. 48.13a,b Comparison of steady-state performance as a function of the delay d_G of $G(q, k)$ ($G = 10$ dB) for the acoustic feedback path with a mobile phone attached to the head. **(a)** Misalignment of CAF, C-CAF, and BL-CAF. **(b)** Misalignment of CAF, PEMAFC-f, and PEMAFC-a. Real speech as a desired signal $x[k]$

48.5.2 Tracking Performance

To illustrate the convergence and tracking performance of the algorithms, an abrupt change of the feedback path was simulated after 10 s by switching the test condition from the scenario without obstruction to the scenario with a mobile phone attached to the head. After 20 s the mobile phone was removed again. The forward path gain and delay were $G = 17$ dB and $d_G = 4$ ms. Figure 48.14a depicts the misalignment of CAF, C-CAF, and BL-CAF as a function of time; Fig. 48.14b depicts the misalignment of CAF, PEMAFC-f, and PEMAFC-a.

When the sudden changes in feedback path occur, the hearing-aid system temporarily becomes unstable. For this scenario, all algorithms succeed in restabilizing the hearing aid after the feedback path changes. The faster the convergence of the algorithm, the shorter the duration of instability. Before 10 s and after 20 s, C-CAF and BL-CAF converge faster than CAF and have a smaller misalignment. Between 10 s and 20 s, the CAF performs quite well. During this time period, the loop gain and hence the energy ratio of the feedback signal to the (disturbing) desired signal is high, explaining the good performance. The PEMAFC methods pre-whiten the desired signal (which acts as a disturbance) to some extent, generally resulting in a faster convergence and hence, better tracking compared to CAF, C-CAF, and BL-CAF. Among all algorithms, PEMAFC-a has the fastest convergence and best tracking performance thanks to the adaptive pre-whitening of the desired signal.

48.5.3 Measurement of the Actual Maximum Stable Gain

To illustrate the effective benefit of the different algorithms for a hearing aid user, the MSG of the algorithms was determined by increasing the forward path gain until audible distortion (such as howling) occurred. (This gain was smaller than the gain at which instability occurred.) The desired signal $x[k]$ was a real speech sig-

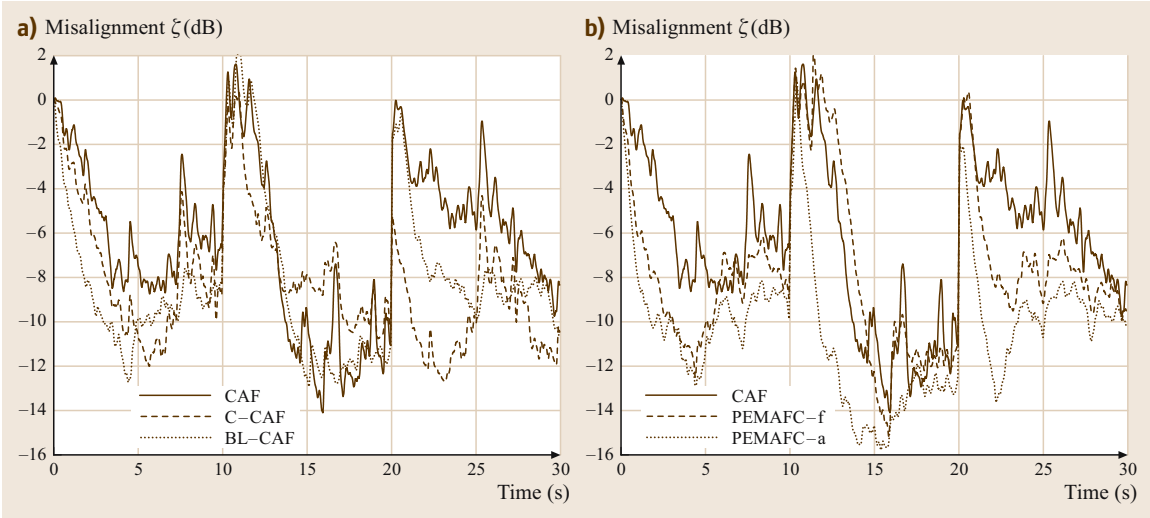


Fig. 48.14a,b Convergence and tracking performance of the various algorithms. At 10 s and 20 s a sudden change in the feedback path occurs. **(a)** Misalignment as a function of time of CAF, C-CAF, and BL-CAF. **(b)** Misalignment as a function of time of CAF, PEMAFC-f, and PEMAFC-a

nal, consisting of sentences spoken by a male speaker. The initial gain in the forward path was set at 6 dB below the stability margin. After 1 s of convergence, the gain was gradually and slowly increased in steps of 0.1 dB each 0.5 s, while the adaptive algorithm was running.

Table 48.3 compares the MSG without feedback cancellation (abbreviated in Table 48.3 as No FC) with the MSG that was achieved with the different algorithms. Three forward path delays d_G are considered, i. e., $d_G = 2$ ms, $d_G = 4$ ms, and $d_G = 6$ ms. For $d_G = 2$ ms, the benefit in MSG by the CAF is small: without obstruction, the MSG even decreases. Increasing the delay d_G significantly improves the performance of the CAF (see Fig. 48.5). Indeed, for $d_G = 4$ ms and 6 ms, the CAF on average increases the MSG of the hearing aid by 13.5 dB (without obstruction) and 19.5 dB (with a mobile phone).

For the scenario without obstruction, the C-CAF increases the MSG of the hearing aid by 15 dB. Especially for $d_G = 2$ ms, the C-CAF outperforms the CAF. For the scenario with a mobile phone, the C-CAF does not achieve better performance than the CAF. For this scenario, the reference filter in the C-CAF significantly deviates from the true feedback path (see Fig. 48.7), explaining the worse performance. The BL-CAF improves the MSG of the hearing aid by 14 dB (without obstruction) and 11 dB (with a mobile phone). For $G > 30$ dB, the frequencies below the cut-off frequency $f_c = 1$ kHz become critical. This explains why the MSG of the

Table 48.3 Maximum stable gain (MSG) in dB achieved with the various algorithms for a real speech signal $x[k]$. For comparison, the MSG achieved without feedback canceller is also shown ('No FC')

	No FC	CAF	C-CAF	BL-CAF	PEMAFC-f	PEMAFC-a
No obstruction						
$d_G = 2$ ms	17 dB	14 dB	32 dB	31 dB	30 dB	41 dB
$d_G = 4$ ms	17 dB	33 dB	33 dB	27 dB	35 dB	37 dB
$d_G = 6$ ms	16 dB	27 dB	31 dB	35 dB	37 dB	39 dB
Mobile phone						
$d_G = 2$ ms	9 dB	16 dB	17 dB	27 dB	33 dB	39 dB
$d_G = 4$ ms	9 dB	31 dB	22 dB	26 dB	38 dB	38 dB
$d_G = 6$ ms	10 dB	27 dB	23 dB	27 dB	37 dB	39 dB

BL-CAF with $f_c = 1\,\text{kHz}$ is limited to about 30 dB. PEMAFC-f and PEMAFC-a increase the MSG of the hearing aid by 17 dB and 22 dB, respectively, for the scenario without obstruction and by 27 dB and 29 dB, respectively, for the scenario with a mobile phone. For all three delays, they outperform the CAF. PEMAFC-a especially offers a benefit compared to PEMAFC-f for low delays d_G.

48.6 Conclusions

This chapter gave an overview of existing feedforward suppression and, in particular, feedback cancellation techniques. The most promising approach to reduce acoustic feedback is the use of a continuous adaptation feedback (CAF) canceller. However, standard continuous adaptation feedback cancellers (CAF) fail to provide a reliable feedback path estimate when the desired signal is spectrally colored.

Existing techniques to reduce the bias of the standard CAF were discussed. Common approaches include introducing signal-decorrelating operations in the forward path or reducing the adaptation speed of the feedback canceller. Other techniques control the adaptation of the feedback canceller based on prior knowledge of the acoustic feedback path through constrained (C-CAF) or bandlimited adaptation (BL-CAF). A final class of techniques views the feedback path as part of a closed-loop system and applies closed-loop system identification using the direct, the indirect or joint input–output method. The direct method seems especially appealing for feedback cancellation, since it relies on neither assumptions about the forward path nor on the use of an external excitation signal. In this method, the bias is reduced by incorporating a stationary (PEMAFC-f) or a time-varying (PEMAFC-a) model of the desired signal.

The performance of CAF, C-CAF, BL-CAF, and PEMAFC was compared for real speech as a desired signal based on acoustic feedback paths measured in a commercial hearing aid. Simulations showed that C-CAF, BL-CAF, and PEMAFC all reduce the bias of the CAF, especially for small processing delays. Among all these methods, PEMAFC with adaptive desired signal model (PEMAFC-a) achieved the best tracking performance and largest maximum stable gain.

References

48.1 U. Forssell: Closed-loop Identification – Methods, Theory and Applications. Ph.D. Thesis (Linköping Universitet, Linköping 1999)

48.2 L. Ljung, T. Söderström: *Theory and Practice of Recursive Identification* (MIT, Cambridge 1983)

48.3 J. Hellgren, T. Lunner, S. Arlinger: System identification of feedback in hearing aids, J. Acoust. Soc. Am. **105**(6), 3481–3496 (1999)

48.4 J. Hellgren, T. Lunner, S. Arlinger: Variations in the feedback of hearing aids, J. Acoust. Soc. Am. **106**(5), 2821–2833 (1999)

48.5 B. Rafaely, M. Roccasalva-Firenze, E. Payne: Feedback path variability modeling for robust hearing aids, J. Acoust. Soc. Am. **107**(5), 2665–2673 (2000)

48.6 M.R. Stinson, G.A. Daigle: Effect of handset proximity on hearing aid feedback, J. Acoust. Soc. Am. **115**(3), 1147–1156 (2004)

48.7 D.K. Bustamante, T.L. Worrall, M.J. Williamson: Measurement and adaptive suppression of acoustic feedback in hearing aids, Proc. ICASSP, Vol. 3 (Glasgow 1989) pp. 2017–2020

48.8 V. Hohmann, V. Hamacher, I. Holube, B. Kollmeier, T. Wittkop: Method for the operation of a hearing aid device or hearing device system as well as hearing aid device or hearing aid device system, US Patent 0176595 (2002), A1

48.9 J.A. Maxwell, P.M. Zurek: Reducing acoustic feedback in hearing aids, IEEE Trans. Speech Audio Process. **3**(4), 304–313 (1995)

48.10 R. Porayath, J. Daniel: Acoustic feedback elimination using adaptive notch filter algorithm, US Patent 5999631 (1999)

48.11 R. Wang, R. Harjani: Acoustic feedback cancellation in hearing aids, Proc. ICASSP, Vol. 1 (1993) pp. 137–140

48.12 J.L. Nielsen, U.P. Svensson: Performance of some linear time-varying systems in control of acoustic feedback, J. Acoust. Soc. Am. **1**(106), 240–254 (1999)

48.13 M.R. Schroeder: Improvement of acoustic-feedback stability by frequency shifting, J. Acoust. Soc. Am. **36**(9), 1718–1724 (1976)

48.14 J.M. Kates: Room reverberation effects in hearing aid feedback cancellation, J. Acoust. Soc. Am. **109**(1), 367–378 (2001)

48.15 H.-F. Chi, S.X. Goa, S.D. Soli, A. Alwan: Band-limited feedback cancellation with a modified

filtered-X LMS algorithm for hearing aids, Speech Commun. **39**(1-2), 147–161 (2003)
48.16 J.E. Greenberg, P.M. Zurek, M. Brantley: Evaluation of feedback-reduction algorithms for hearing aids, J. Acoust. Soc. Am. **108**(5), 2366–2376 (2000)
48.17 N.A. Shusina, B. Rafaely: Unbiased adaptive feedback cancellation in hearing aids by closed-loop identification, IEEE Trans. Audio Speech Lang. Process. **14**(2), 658–665 (2006)
48.18 A. Spriet, I. Proudler, M. Moonen, J. Wouters: Adaptive feedback cancellation in hearing aids with linear prediction of the desired signal, IEEE Trans. Signal Process. **53**(10), 3749–3763 (2005)
48.19 J.M. Kates: Feedback cancellation in hearing aids: Results from a computer simulation, Signal Process. **39**(3), 553–562 (1991)
48.20 Y. Park, D. Kim, I. Kim: An efficient feedback cancellation for multiband compression hearing aids, Proc. 20th Annu. Int. Conf. Eng. Med. Biol. Soc., Vol. 5 (1998) pp. 2706–2709
48.21 S. Thipphayathetthana, C. Chinrungrueng: Variable step-size of the least-mean-square algorithm for reducing acoustic feedback in hearing aids, IEEE Asi-Pacific Conf. on Circuits and Systems (2000) pp. 407–410
48.22 J.M. Kates: Adaptive feedback cancellation in hearing aids. In: *Adaptive Signal Processing: Applications to Real-World Problems*, ed. by J. Benesty, Y. Huang (Springer, Berlin, Heidelberg 2003) pp. 23–55
48.23 J.E. Greenberg: Modified LMS algorithms for speech processing with an adaptive noise canceller, IEEE Trans. Speech Audio Process. **6**(4), 338–350 (1998)
48.24 T. Fillon, J. Prado: Acoustic feedback cancellation for hearing-aids, using multi-delay filter, 5th Nordic Signal Process. Symp. (NORSIG) (on board Hurtigruten, 2002)
48.25 A. Kaelin, A. Lindgren, S. Wyrsch: A digital frequency-domain implementation of a very high gain hearing aid with compensation for recruitment of loudness and acoustic echo cancellation, Signal Process. **64**(1), 71–85 (1998)
48.26 J.-S. Soo, K. Pang: Multidelay block frequency domain adaptive filter, IEEE Trans. Acoust. Speech Signal Process. **38**(4), 788–798 (1990)
48.27 A. Spriet, G. Rombouts, M. Moonen, J. Wouters: Adaptive feedback cancellation in hearing aids, J. Franklin Inst. *343*, 545–572 (2006)
48.28 M.G. Siqueira, A. Alwan: Steady-state analysis of continuous adaptation in acoustic feedback reduction systems for hearing-aids, IEEE Trans. Speech Audio Process. **8**(4), 443–453 (2000)
48.29 M.A. Stone, B.C.J. Moore: Tolerable hearing aid delays. I. estimation of limits imposed by the auditory path alone using simulated hearing losses, Ear Hear. **20**(3), 182–191 (1999)
48.30 J. Hellgren: Analysis of feedback cancellation in hearing aids with filtered-X LMS and the direct method of closed loop identification, IEEE Trans. Speech Audio Process. **10**(2), 119–131 (2002)
48.31 S. Laugesen, K.V. Hansen, J. Hellgren: Acceptable delays in hearing aids and implications for feedback cancellation, J. Acoust. Soc. Am. **105**(2), 1211–1212 (1999)
48.32 M. Nilsson, S.D. Soli, A. Sullivan: Development of the Hearing in Noise Test for the measurement of speech reception thresholds in quiet and in noise, J. Acoust. Soc. Am. **95**(2), 1085–1099 (1994)
48.33 A.M. Engebretson, M.F. George: Properties of an adaptive feedback equalization algorithm, J. Rehabil. Res. Devel. **30**(1), 8–16 (1993)
48.34 C.P. Janse, P.A.A. Timmermans: Signal amplifier system with improved echo cancellation, US Patent 5748751 (1998)
48.35 H.A.L. Joson, F. Asano, Y. Suzuki, S. Toshio: Adaptive feedback cancellation with frequency compression for hearing aids, J. Acoust. Soc. Am. **94**(6), 3248–3254 (1993)
48.36 T. Kaulberg: A hearing aid with an adaptive filter for suppression of acoustic feedback, European Patent 1191814 (2002)
48.37 J. Nielsen, M. Ekelid: Feedback cancellation using bandwidth detection, W0 Patent 01/06746*A2* (2001)
48.38 J. Nielsen, M. Ekelid: Feedback cancellation with low frequency input, W0 Patent 01/06812*A1* (2001)
48.39 J.M. Kates: Constrained adaptation for feedback cancellation in hearing aids, J. Acoust. Soc. Am. **106**(2), 1010–1019 (1999)
48.40 S. Gao, S. Soli, H.-F. Chi: Band-limited adaptive feedback canceller for hearing aids, European Patent 1118247 (1999)
48.41 B. Rafaely, N.A. Shusina, J.L. Hayes: Robust compensation with adaptive feedback cancellation in hearing aids, Speech Commun. **39**(1-2), 163–170 (2003)
48.42 S.D. Soli, P. Widin, K.M. Buckley: Auditory prosthesis, noise suppression apparatus and feedback suppression apparatus having focused adaptive filtering, US Patent 5402496 (1993)
48.43 A. Spriet, I. Proudler, M. Moonen, J. Wouters: An instrumental variable method for adaptive feedback cancellation in hearing aids, Proc. ICASSP, Vol. 3 (Philadelphia 2005) pp. 129–132
48.44 U. Forssell, L. Ljung: Closed-loop identification revisited, Automatica **35**, 1215–1241 (1999)
48.45 J. Hellgren, U. Forssell: Bias of feedback cancellation algorithms in hearing aids based on direct closed loop identification, IEEE Trans. Speech Audio Process. **9**(7), 906–913 (2001)
48.46 R. Leber, W. Schaub: Circuit and method for the adaptive suppression of an acoustic feedback, US Patent 6611600 (2003)

48.47 N.A. Shusina, B. Rafaely: Feedback cancellation in hearing aids based on indirect closed-loop identification, Proc. IEEE Benelux Signal Process. Symp. (SPS) (Leuven 2002) pp. 177–180

48.48 G. Rombouts, T. van Waterschoot, K. Struyve, M. Moonen: Acoustic feedback suppression for long acoustic paths using a nonstationary source model, IEEE Trans. Signal Process. **54**(9), 3426–3434 (2004), accepted for publication, Available as Technical Report ESAT-SISTA/TR 2004-151, K.U.

48.49 J.R. Deller, J.G. Proakis, J.H.L. Hansen: *Discrete-Time Processing of Speech Signals* (Macmillan, Englewood Cliffs 1993)

48.50 N. Bisgaard, O. Dyrlund: Acoustic feedback part 2: A digital system for suppression of feedback, Hear. Instrum. **42**, 44–45 (1991)

48.51 H.R. Skovgaard: Hearing aid compensating for acoustic feedback, US Patent 5680467 (1997)

48.52 J.H. Stott, N.D. Wells: Method and apparatus for reduction of unwanted feedback, US Patent 6269165 (2001)

43. Fundamentals of Noise Reduction

J. Chen, J. Benesty, Y. Huang, E. J. Diethorn

The existence of noise is inevitable. In all applications that are related to voice and speech, from sound recording, telecommunications, and telecollaborations, to human–machine interfaces, the signal of interest that is picked up by a microphone is generally contaminated by noise. As a result, the microphone signal has to be *cleaned up* with digital signal-processing tools before it is stored, analyzed, transmitted, or played out. The cleaning process, which is often referred to as either noise reduction or speech enhancement, has attracted a considerable amount of research and engineering attention for several decades. Remarkable advances have already been made, and this area is continuing to progress, with the aim of creating processors that can extract the desired speech signal as if there is no noise. This chapter presents a methodical overview of the state of the art of noise-reduction algorithms. Based on their theoretical origin, the algorithms are categorized into three fundamental classes: filtering techniques, spectral restoration, and model-based methods. We outline the basic ideas underlying these approaches, discuss their characteristics, explain their intrinsic relationships, and review their advantages and disadvantages.

43.1 Noise

Since we live in a natural environment where noise is inevitable and ubiquitous, speech signals are generally immersed in noise and can seldom be acquired and processed in pure form. Noise can profoundly affect human-to-human and human-to-machine communications, including changing a talker's speaking pattern, modifying the characteristics of the speech signal, degrading speech quality and intelligibility, and affecting the listener's perception and machine's processing of the recorded speech. In order to make voice communication feasible, natural, and comfortable in the presence of noise regardless of the noise level, it is desirable to develop digital signal processing techniques to *clean* the microphone signal before it is stored,

transmitted, or played out. This problem has been a major challenge for many researchers and engineers for decades [43.1].

Generally speaking, noise is a term used to signify any unwanted signal that interferes with measurement, processing, and communication of the desired information-bearing speech signal. This broad-sense definition, however, is too encompassing as it masks many important technical aspects of the real problem. To enable better modeling and removal of the effects of noise, it is advantageous to break the general definition of noise into the following four subcategories: *additive noise* originating from various ambient sound sources, *interfering signals* from concurrent competing speakers, *reverberation* caused by multipath propagation introduced by an enclosure, and *echo* resulting from coupling between loudspeakers and microphones. Combating these four problems has led to the developments of diverse acoustic signal processing techniques. They include noise reduction (or speech enhancement), source separation, speech dereverberation, and echo cancelation and suppression, each of which is a rich subject of research. This chapter will focus on techniques to eliminate or mitigate the effects of additive noise, while the approaches to the remaining three problems will be addressed in other chapters in Parts H and I. So, from now on, we shall narrow the definition of noise to additive noise.

Under this narrowed definition, the observed microphone signal can be modeled as a superposition of the clean speech and noise. The objective of noise reduction, then, becomes to restore the original clean speech from the mixed signal. This can be described as a parameter estimation problem and the optimal estimate of the clean speech can be achieved by optimizing some criterion, such as mean-square error (MSE) between the clean speech and its estimate, signal-to-noise ratio (SNR), the a posteriori probability of the clean speech given its noisy observations, etc. Unfortunately, an optimal estimate formed from a signal processing perspective does not necessarily correspond to the best quality when perceived by the human ear. This inconsistency between objective measures and subjective quality judgement has forced researchers to rethink performance criteria for noise reduction. The objective of the problem has subsequently been broadened, which can be summarized as to achieve one or more of the following three primary goals:

1. to improve objective performance criteria such as intelligibility, SNR, etc.;
2. to improve the perceptual quality of degraded speech; and
3. as a preprocessor, to increase robustness of other speech processing applications (such as speech coding, echo cancelation, automatic speech recognition, etc.) to noise.

The choice of a different goal will lead to a distinct speech estimate. It is very hard to satisfy all three goals at the same time.

With a specified goal (performance criterion), the difficulty and complexity of the noise-reduction problem can vary tremendously, depending on the number of microphone channels. In general, the larger the number of microphones, the easier the noise-reduction problem. For example, when an array of microphones can be used, a beam can be formed and steered to a desired direction. As a result, signals propagating from the desired direction will be passed through without degradation, while signals originating from all other directions will either suffer a certain amount of attenuation or be completely rejected [43.2–4]. In the two-microphone case, with one microphone picking up the noisy signal and the other measuring the noise field, we can use the second microphone signal as a noise reference and eliminate the noise in the first microphone by means of adaptive cancelation [43.5–9]. However, most of today's communication terminals are equipped with only one microphone. In this case, noise reduction becomes a very difficult problem since no reference of the noise is accessible and the clean speech cannot be preprocessed prior to being corrupted by noise. Due to its wide range of potential applications, however, this single-channel noise-reduction problem has attracted a significant amount of research attention, and is also the focus of this chapter.

Pioneering research on the single-channel noise reduction was started more than 40 years ago with two patents by *Schroeder* [43.10, 11]. He proposed an analog implementation of spectral magnitude subtraction. This work, however, has not received much public attention, probably because it was never published in journals or conferences. About 15 years later, *Boll*, in his informative paper [43.12], reinvented the spectral subtraction method but in the digital domain. Almost at the same time, *Lim* and *Oppenheim*, in their landmark work [43.13], systematically formulated the noise-reduction problem and studied and compared the different algorithms known at that time. Their work demonstrated that noise reduction is not only useful in improving the quality of noise-corrupted speech,

but also useful in increasing both the quality and intelligibility of linear prediction coding (LPC)-based parametric speech coding systems. It is this work that has sparkled a huge amount of research attention on the problem.

By and large, the developed techniques can be broadly classified into three categories, i. e., filtering technique, spectral restoration, and model-based methods. The basic principle underlying the filtering technique is to pass the noisy speech through a linear filter/transformation. Since speech and noise normally have very different characteristics, the filter/transformation can be designed to significantly attenuate the noise level while leaving the clean speech relatively unchanged. The representative algorithms in this category include Wiener filters [43.12–18], subspace methods [43.19–26], etc. Comparatively, the spectral restoration technique treats noise reduction as a robust spectral estimation problem, i. e., estimating the spectrum of the clean speech from that of the noise-corrupted speech. Many algorithms have been developed for this purpose, such as the minimum-mean-square-error (MMSE) estimator [43.27,28], the maximum-likelihood (ML) estimator [43.29], and the maximum a posteriori (MAP) estimator [43.30], to name a few. Similar to the spectral restoration technique, these model-based approaches also formulate noise reduction as a parameter estimation problem. The difference between the two is that, in the model-based methods, a mathematical model is used to represent human speech production and parameter estimation is carried out in the model space, which normally has a much lower dimensionality than the original signal space. Typical algorithms in this group include harmonic-model-based methods [43.13, 14], linear-prediction-model-based Kalman filtering approaches [43.31–33], and hidden-Markov-model-based statistical methods [43.34–36]. This chapter attempts to briefly summarize all these techniques and review recent advances that have significantly improved the performance of noise reduction. We will outline the basic ideas underlying these approaches, analyze their performance, discuss their intrinsic relationships, and review their advantages and disadvantages.

43.2 Signal Model and Problem Formulation

The noise-reduction problem considered in this chapter is to recover a speech signal of interest $x(n)$ from the noisy observation

$$y(n) = x(n) + v(n)\,, \tag{43.1}$$

where $v(n)$ is the unwanted additive noise, which is assumed to be a zero-mean random process (white or colored) and uncorrelated with $x(n)$.

In vector/matrix form, the signal model (43.1) is written as

$$\boldsymbol{y}(n) = \boldsymbol{x}(n) + \boldsymbol{v}(n)\,, \tag{43.2}$$

where

$$\boldsymbol{y}(n) = [\; y(n) \;\; y(n-1) \;\; \cdots \;\; y(n-L+1) \;]^{\mathrm{T}}\,,$$

is a vector consisting of the L most-recent samples of the noisy speech signal, superscript $^{\mathrm{T}}$ denotes transpose of a vector or a matrix, and $\boldsymbol{x}(n)$ and $\boldsymbol{v}(n)$ are defined in a similar way to $\boldsymbol{y}(n)$. In this case, the noise-reduction problem is formulated as one of estimating $\boldsymbol{x}(n)$ from the observation $\boldsymbol{y}(n)$.

Applying an L-point discrete Fourier transform (DFT) to both sides of (43.2), we have the following relationship in the frequency domain:

$$Y(n, \mathrm{i}\omega_k) = X(n, \mathrm{i}\omega_k) + V(n, \mathrm{i}\omega_k)\,, \tag{43.3}$$

where

$$Y(n, \mathrm{i}\omega_k) = \sum_{l=0}^{L-1} w(l) y(n-L+l+1) \mathrm{e}^{-\mathrm{i}\omega_k l}\,,$$

is the short-time DFT of the noisy speech at time instant n, $\omega_k = 2\pi k/L$, $k = 0, 1, \cdots, L-1$, $w(l)$ is a window function (e.g., Hamming window, Hann window), and $X(n, \mathrm{i}\omega_k)$ and $V(n, \mathrm{i}\omega_k)$ are the short-time DFTs of the clean speech and noise signals, defined in a similar way to $Y(n, \mathrm{i}\omega_k)$. Based on this relationship, the noise-reduction problem can be expressed in the frequency domain as one of estimating $X(n, \mathrm{i}\omega_k)$ from $Y(n, \mathrm{i}\omega_k)$.

43.3 Evaluation of Noise Reduction

The goal of noise reduction is to attenuate the background noise while keeping the desired speech signal unchanged. In order to verify whether this goal has been fulfilled after application of an algorithm, we need some performance criterion. Generally speaking, there are two categories of performance measures, i. e., subjective and objective ones. Subjective measures rely on human listeners' judgements and are typically used for evaluating speech quality. Commonly used subjective tests include a so-called naturalness test [43.37], mean-opinion-score (MOS) tests [43.37, 38], categorical estimation (CE) [43.39], and magnitude estimation [43.39]. In contrast, objective measures, often derived either from pure signal processing considerations or from combinations of signal processing and human speech perception, are independent of listeners' preferences. As we mentioned earlier, one major goal of noise reduction is to improve the quality of degraded speech. As far as speech quality is concerned, the subjective method should be the most appropriate performance criterion because it is the listener's judgment that ultimately counts. However, subjective evaluation is often labor intensive and time consuming and the results are expensive to obtain. In addition, it may not be able to provide timely guidance on algorithm optimization. Comparatively, objective measures are often economic to use. Moreover, some objective criteria can also be integrated into optimization cost functions to guide algorithm design. For these reasons, great efforts have been devoted to the development of objective measures [43.40]. Some widely accepted and used objective measures include signal-to-noise ratio (SNR), log-spectral distance [43.41, 42], Itakura distance, and Itakura–Saito distance [43.40, 43, 44]. Before discussing noise-reduction algorithms, we list some objective measures that are helpful in understanding the performance behavior of noise-reduction algorithms.

43.3.1 Signal-to-Noise Ratio

Signal-to-noise ratio (SNR) is one of the most often used terms in the field of noise reduction; it quantifies how noisy a signal is. This ratio is defined as the signal intensity relative to the intensity of additive background noise and is usually measured in decibels (dB). With the signal model shown in (43.1), the SNR of the microphone signal $y(n)$ is given by

$$\mathrm{SNR} \triangleq \frac{\sigma_x^2}{\sigma_v^2} = \frac{E[x^2(n)]}{E[v^2(n)]}\,, \tag{43.4}$$

where $E[\cdot]$ denotes statistical expectation. A frequency-domain counterpart of the above definition can also be formulated according to the well-known Parseval's relation

$$\mathrm{SNR} = \frac{\int_{-\pi}^{\pi} P_x(\omega)\,\mathrm{d}\omega}{\int_{-\pi}^{\pi} P_v(\omega)\,\mathrm{d}\omega}\,, \tag{43.5}$$

where $P_x(\omega)$ and $P_v(\omega)$ are, respectively, the power spectral densities of the signal $x(n)$ and noise $v(k)$, and ω is the angular frequency. Apparently, the higher the SNR value, the cleaner the signal.

In the context of noise reduction, both the terms a priori SNR and a posteriori SNR are often used. The former usually refers to the SNR of the observed noisy signal, and the latter regards the SNR of the noise-reduced signal. The difference between the two is called the SNR improvement. The higher the SNR improvement, the more effective the noise-reduction algorithm.

43.3.2 Noise-Reduction Factor and Gain Function

The primary issue that we must determine with noise reduction is how much noise is actually attenuated. The noise-reduction factor is a measure of this, and is defined as the ratio between the original noise intensity and the intensity of the residual noise remaining in the noise-reduced speech. Consider the observed signal given in (43.1). If, after noise reduction, the residual noise is denoted by $v_\mathrm{r}(n)$, the noise-reduction factor is mathematically written as

$$\xi_\mathrm{nr} \triangleq \frac{E[v^2(n)]}{E\left[v_\mathrm{r}^2(n)\right]}\,. \tag{43.6}$$

This factor is greater than one when noise is indeed reduced. The larger the value of ξ_nr, the greater the noise reduction.

It is known that a speech waveform consists of a sequence of different events. Its characteristics, therefore, fluctuate over time and frequency. Often, the characteristics of ambient noise also vary across frequency bands. This indicates that we cannot achieve uniform noise-reduction performance over the frequency range of interest; rather, it changes from one frequency band to another. Although the noise-reduction factor gives us an idea of the overall noise attenuation, it does not tell us the noise-reduction behavior from frequency to frequency. A more-useful and insightful frequency-dependent measure of noise reduction can be defined as

the ratio between the spectral densities of the original and residual noise, i. e.,

$$\Psi_{\mathrm{nr}}(\omega) \triangleq \frac{E[|V(\mathrm{i}\omega)|^2]}{E\left[|V_{\mathrm{r}}(\mathrm{i}\omega)|^2\right]} = \frac{P_v(\omega)}{P_{v_{\mathrm{r}}}(\omega)} , \tag{43.7}$$

where $V(\mathrm{i}\omega)$, $V_{\mathrm{r}}(\mathrm{i}\omega)$, $P_v(\omega)$, and $P_{v_{\mathrm{r}}}(\omega)$ are the Fourier spectra and power spectral densities of $v(k)$ and $v_{\mathrm{r}}(k)$, respectively. This measure reflects the noise-reduction gain as a function of frequency. It is therefore logical to call it the noise-reduction gain function.

43.3.3 Speech-Distortion Index and Attenuation Frequency Distortion

Noise reduction is often achieved at a price of speech distortion. The degree of speech distortion is a very complicated issue that depends on many factors, such as the amount of a priori knowledge about the speech and noise that can be accessed. The speech-distortion index is a measure that was introduced to quantify the speech distortion due to a noise-reduction algorithm. With the signal model given in (43.1), if we denote the speech component in the speech estimate as $\hat{x}_{\mathrm{nr}}(n)$, the speech-distortion index is defined as [43.18]

$$\varphi_{\mathrm{sd}} \triangleq \frac{E\left\{\left[x(n)-\hat{x}_{\mathrm{nr}}(n)\right]^2\right\}}{E[x^2(n)]} . \tag{43.8}$$

This parameter is lower bounded by 0 and expected to be upper bounded by 1. A larger value of φ_{sd} means more speech distortion. This index, therefore, should be kept as small as possible.

To measure speech distortion over frequency, we can borrow the concept of attenuation frequency distortion from telecommunication theory. The attenuation frequency distortion is a measure that was developed to assess how a telephone channel preserves the fidelity of a speech signal. It is defined as the change in amplitude of the transmitted signal over a voice band. Here we adjust this concept and define the attenuation frequency distortion for noise reduction as

$$\begin{aligned}\Phi_{\mathrm{sd}}(\omega) &\triangleq \frac{E\left[|X(\mathrm{i}\omega)|^2 - |\hat{X}_{\mathrm{nr}}(\mathrm{i}\omega)|^2\right]}{E[|X(\mathrm{i}\omega)|^2]} \\ &= \frac{P_x(\omega) - P_{\hat{x}_{\mathrm{nr}}}(\omega)}{P_x(\omega)} ,\end{aligned} \tag{43.9}$$

where $X(\mathrm{i}\omega)$ and $P_x(\omega)$ are the Fourier spectrum and power spectral density of the clean speech $x(n)$, and $\hat{X}_{\mathrm{nr}}(\mathrm{i}\omega)$ and $P_{\hat{x}_{\mathrm{nr}}}(\omega)$ are, respectively, the Fourier spectrum and power spectral density of the speech component in the noise-reduced signal. Note that whereas (43.8) is a *coherent* measure of distortion, (43.9) is an *incoherent* measure. Therefore, Φ_{sd} is not simply a frequency-domain version of φ_{sd}; they are fundamentally different.

43.4 Noise Reduction via Filtering Techniques

We are now ready to discuss noise-reduction algorithms. Let us first consider filtering techniques. The basic principle behind this class of techniques is to design a linear filter/transformation so that, when the noisy speech is passed through such a filter/transformation, the noise component will be attenuated. The representative algorithms include the time- and frequency-domain Wiener filters, the subspace method, and the parametric Wiener filter.

43.4.1 Time-Domain Wiener Filter

The Wiener filter is one of the most fundamental approaches for noise reduction and can be formulated in either the time or frequency domains. The time-domain Wiener filter is obtained by minimizing the mean-square error (MSE) between the signal of interest and its estimate.

With the signal model given in (43.1), an estimate of the clean speech can be obtained by passing the noisy signal $y(n)$ through a temporal filer, i. e.,

$$\hat{x}(n) = \boldsymbol{h}^{\mathrm{T}}\boldsymbol{y}(n) , \tag{43.10}$$

where

$$\boldsymbol{h} = [\, h_0 \;\; h_1 \;\; \cdots \;\; h_{L-1} \,]^{\mathrm{T}} \tag{43.11}$$

is a finite impulse response (FIR) filter of length L. The error signal between the clean speech sample and its estimate at time n is defined as

$$e_x(n) \triangleq x(n) - \hat{x}(n) = x(n) - \boldsymbol{h}^{\mathrm{T}}\boldsymbol{y}(n) . \tag{43.12}$$

The MSE criterion is then written as

$$J_x(\boldsymbol{h}) \triangleq E\left\{e_x^2(n)\right\} . \tag{43.13}$$

Consider the particular filter,

$$\boldsymbol{h}_1 = [\, 1 \;\; 0 \;\; \cdots \;\; 0 \,]^{\mathrm{T}} . \tag{43.14}$$

This means that the observed signal $y(n)$ will pass this filter unaltered (no noise reduction). Thus the corresponding MSE is

$$J_x(\boldsymbol{h}_1) = E\{v^2(n)\} = \sigma_v^2 \,. \tag{43.15}$$

The optimal estimate $\hat{x}_o(n)$ of the clean speech sample $x(n)$ tends to contain less noise than the observation sample $y(n)$, and the optimal filter that forms $\hat{x}_o(n)$ is the Wiener filter, which is obtained as

$$\boldsymbol{h}_o = \arg\min_{\boldsymbol{h}} J_x(\boldsymbol{h}) \,. \tag{43.16}$$

In principle, for the optimal filter $\boldsymbol{h}_o$, we should have

$$J_x(\boldsymbol{h}_o) < J_x(\boldsymbol{h}_1) = \sigma_v^2 \,. \tag{43.17}$$

In other words, the Wiener filter should be able to reduce the level of noise in the noisy speech signal $y(n)$.

From (43.16), we easily find the Wiener–Hopf equations

$$\boldsymbol{R}_y \boldsymbol{h}_o = \boldsymbol{r}_{yx} \,, \tag{43.18}$$

where

$$\boldsymbol{R}_y = E\{\boldsymbol{y}(n)\boldsymbol{y}^T(n)\} \tag{43.19}$$

is the correlation matrix of the observed signal $y(n)$ and

$$\boldsymbol{r}_{yx} = E\{\boldsymbol{y}(n)x(n)\} \tag{43.20}$$

is the cross-correlation vector between the noisy and clean speech signals.

It is seen from (43.18) that both $\boldsymbol{R}_y$ and $\boldsymbol{r}_{yx}$ need to be known in order to compute the Wiener filter $\boldsymbol{h}_o$. The correlation matrix $\boldsymbol{R}_y$ can be directly estimated from the observation signal $y(n)$. However, $x(n)$ is unobservable; as a result, an estimate of $\boldsymbol{r}_{yx}$ may seem difficult to obtain. But since speech and noise are uncorrelated, we can readily derive

$$\boldsymbol{r}_{yx} = E\{\boldsymbol{y}(n)y(n)\} - E\{\boldsymbol{v}(n)v(n)\} = \boldsymbol{r}_{yy} - \boldsymbol{r}_{vv} \,. \tag{43.21}$$

Now $\boldsymbol{r}_{yx}$ depends on two correlation vectors: $\boldsymbol{r}_{yy}$ and $\boldsymbol{r}_{vv}$. The vector $\boldsymbol{r}_{yy}$, which is the first column of $\boldsymbol{R}_y$, can also be directly estimated from $y(n)$. The vector $\boldsymbol{r}_{vv}$ can be measured during intervals where the speech signal is absent. As a result, the Wiener–Hopf equations given in (43.18) can be rewritten as

$$\boldsymbol{R}_y \boldsymbol{h}_o = \boldsymbol{r}_{yy} - \boldsymbol{r}_{vv} \,. \tag{43.22}$$

If we assume that the matrix $\boldsymbol{R}_y$ is full rank, which is the case in most practical situations, the Wiener filter can be obtained by solving either (43.18) or (43.22), i. e.,

$$\boldsymbol{h}_o = \boldsymbol{R}_y^{-1}\boldsymbol{r}_{yx} = \boldsymbol{R}_y^{-1}\boldsymbol{r}_{yy} - \boldsymbol{R}_y^{-1}\boldsymbol{r}_{vv} = \boldsymbol{h}_1 - \boldsymbol{R}_y^{-1}\boldsymbol{r}_{vv} \,. \tag{43.23}$$

If we define two normalized correlation matrices

$$\tilde{\boldsymbol{R}}_x \triangleq \frac{\boldsymbol{R}_x}{\sigma_x^2} \,, \quad \tilde{\boldsymbol{R}}_v \triangleq \frac{\boldsymbol{R}_v}{\sigma_v^2} \,, \tag{43.24}$$

where $\boldsymbol{R}_x$ and $\boldsymbol{R}_v$ are, respectively, the correlation matrices of the clean speech and noise, which are defined similarly to $\boldsymbol{R}_y$ in (43.19), the Wiener filter in (43.23) can be further expressed as

$$\begin{aligned}\boldsymbol{h}_o &= \left[\boldsymbol{I} - \boldsymbol{R}_y^{-1}\boldsymbol{R}_v\right]\boldsymbol{h}_1 \\ &= \left[\boldsymbol{I} - (\mathrm{SNR}\cdot\tilde{\boldsymbol{R}}_x + \tilde{\boldsymbol{R}}_v)^{-1}\tilde{\boldsymbol{R}}_v\right]\boldsymbol{h}_1 \,,\end{aligned}$$

where $\boldsymbol{I}$ is the identity matrix and SNR is the signal-to-noise ratio of the microphone signal as defined in (43.4).

Now, as SNR approaches infinity, we have

$$\lim_{\mathrm{SNR}\to\infty} \boldsymbol{h}_o = \boldsymbol{h}_1 \,. \tag{43.25}$$

This is expected, since noise reduction is not needed in this case. At the other extreme, where SNR approaches 0, we have

$$\lim_{\mathrm{SNR}\to 0} \boldsymbol{h}_o = \boldsymbol{0} \,, \tag{43.26}$$

where the vector $\boldsymbol{0}$ has the same size as $\boldsymbol{h}_o$ and consists of all zeros. So, in the absence of any speech signal, the Wiener filter passes nothing.

Now we are ready to check whether the Wiener filter can actually reduce the level of noise as expected. To do this, let us examine the noise-reduction factor. Substituting $\boldsymbol{h}_o$ into (43.10), we obtain the optimal speech estimate

$$\hat{x}_o(n) = \boldsymbol{h}_o^T\boldsymbol{y}(n) = \boldsymbol{h}_o^T\boldsymbol{x}(n) + \boldsymbol{h}_o^T\boldsymbol{v}(n) \,. \tag{43.27}$$

It is seen that there are two terms in the right-hand side of (43.27), where the first, $\boldsymbol{h}_o^T\boldsymbol{x}(n)$, is the clean speech filtered by the Wiener filter, and the second, $\boldsymbol{h}_o^T\boldsymbol{v}(n)$, represents the residual noise. So, the noise-reduction factor, according to (43.6), can be written as

$$\xi_{\mathrm{nr}}(\boldsymbol{h}_o) = \frac{E\left\{\left[\boldsymbol{h}_1^T\boldsymbol{v}(n)\right]^2\right\}}{E\left\{\left[\boldsymbol{h}_o^T\boldsymbol{v}(n)\right]^2\right\}} = \frac{\boldsymbol{h}_1^T\boldsymbol{R}_v\boldsymbol{h}_1}{\boldsymbol{h}_o^T\boldsymbol{R}_v\boldsymbol{h}_o} \,. \tag{43.28}$$

Substituting $\boldsymbol{h}_o = \boldsymbol{R}_y^{-1}\boldsymbol{r}_{yx} = \boldsymbol{R}_y^{-1}\boldsymbol{r}_{xx} = \boldsymbol{R}_y^{-1}\boldsymbol{R}_x\boldsymbol{h}_1$ into (43.28), we get

$$\xi_{\mathrm{nr}}(\boldsymbol{h}_o) = \frac{\boldsymbol{h}_1^T\boldsymbol{R}_v\boldsymbol{h}_1}{\left(\boldsymbol{h}_1^T\boldsymbol{R}_x\boldsymbol{R}_y^{-1}\right)\boldsymbol{R}_v\left(\boldsymbol{R}_y^{-1}\boldsymbol{R}_x\boldsymbol{h}_1\right)} \,, \tag{43.29}$$

which is a function of all the three correlation matrices $\boldsymbol{R}_x$, $\boldsymbol{R}_v$, and $\boldsymbol{R}_y$. Using the generalized eigenvalue

decomposition [43.45], we can decompose the three correlation matrices into the following form:

$$\boldsymbol{R}_x = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{\Lambda}\boldsymbol{B}\,, \tag{43.30a}$$

$$\boldsymbol{R}_v = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{B}\,, \tag{43.30b}$$

$$\boldsymbol{R}_y = \boldsymbol{B}^{\mathrm{T}}(\boldsymbol{I}+\boldsymbol{\Lambda})\boldsymbol{B}\,, \tag{43.30c}$$

where $\boldsymbol{B}$ is an invertible square matrix, and

$$\boldsymbol{\Lambda} = \mathrm{diag}\left(\lambda_1 \;\; \lambda_2 \;\; \cdots \;\; \lambda_L\right) \tag{43.31}$$

is a diagonal matrix with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_L \geq 0$. Substituting (43.30c) into (43.29), we can deduce that

$$\xi_{\mathrm{nr}}(\boldsymbol{h}_{\mathrm{o}}) = \frac{\sum\limits_{i=1}^{L} b_{i1}^2}{\sum\limits_{i=1}^{L} \frac{\lambda_i^2}{(1+\lambda_i)^2} b_{i1}^2}\,, \tag{43.32}$$

where b_{i1}, $i = 1, 2, \cdots, L$, forms the first column of $\boldsymbol{B}$ and satisfies $\sum_{i=1}^{L} b_{i1}^2 = \sigma_v^2$.

Also, with the matrix decomposition in (43.30c), the SNR of the observation signal can be expressed as

$$\mathrm{SNR} = \frac{\boldsymbol{h}_1^{\mathrm{T}}\boldsymbol{R}_x\boldsymbol{h}_1}{\boldsymbol{h}_1^{\mathrm{T}}\boldsymbol{R}_v\boldsymbol{h}_1} = \frac{\sum\limits_{i=1}^{L} \lambda_i b_{i1}^2}{\sum\limits_{i=1}^{L} b_{i1}^2}\,. \tag{43.33}$$

Using (43.33), we can rewrite (43.32) as

$$\begin{aligned}
\xi_{\mathrm{nr}}(\boldsymbol{h}_{\mathrm{o}}) &= \frac{1}{\mathrm{SNR}} \frac{\sum\limits_{i=1}^{L} \lambda_i b_{i1}^2}{\sum\limits_{i=1}^{L} \frac{\lambda_i^2}{(1+\lambda_i)^2} b_{i1}^2} \\
&= \frac{1}{\mathrm{SNR}} \frac{\sum\limits_{i=1}^{L} \frac{(1+\lambda_i)^2}{(1+\lambda_i)^2} \lambda_i b_{i1}^2}{\sum\limits_{i=1}^{L} \frac{\lambda_i^2}{(1+\lambda_i)^2} b_{i1}^2} \\
&= \frac{1}{\mathrm{SNR}} \left(\frac{\sum\limits_{i=1}^{L} \frac{\lambda_i+\lambda_i^3}{(1+\lambda_i)^2} b_{i1}^2}{\sum\limits_{i=1}^{L} \frac{\lambda_i^2}{(1+\lambda_i)^2} b_{i1}^2} + 2 \right)\,.
\end{aligned} \tag{43.34}$$

Using the fact that $\lambda_i + \lambda_i^3 \geq \lambda_i^3$, we easily deduce from (43.34) that

$$\xi_{\mathrm{nr}}(\boldsymbol{h}_{\mathrm{o}}) \geq \frac{1}{\mathrm{SNR}} \left(\frac{\sum\limits_{i=1}^{L} \frac{\lambda_i^3}{(1+\lambda_i)^2} b_{i1}^2}{\sum\limits_{i=1}^{L} \frac{\lambda_i^2}{(1+\lambda_i)^2} b_{i1}^2} + 2 \right)\,. \tag{43.35}$$

Now, before we continue with our discussion on the noise-reduction factor, we first give a lemma.

Lemma 1. With λ_i ($i = 1, 2, \cdots, L$ and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_L \geq 0$) being defined in (43.30c) and $\mu > 0$, we have

$$\begin{aligned}
&\left[\sum_{i=1}^{L} \frac{\lambda_i^3}{(\lambda_i+\mu)^2} q_i^2\right] \sum_{i=1}^{L} q_i^2 \\
&\geq \left[\sum_{i=1}^{L} \frac{\lambda_i^2}{(\lambda_i+\mu)^2} q_i^2\right] \sum_{i=1}^{L} \lambda_i q_i^2\,,
\end{aligned} \tag{43.36}$$

where q_i can be any real numbers.

Proof. This inequality can be proved by way of induction.

- Basic Step: If $L = 2$,

$$\begin{aligned}
&\left(\sum_{i=1}^{2} \frac{\lambda_i^3}{(\lambda_i+\mu)^2} q_i^2\right) \sum_{i=1}^{2} q_i^2 \\
&= \frac{\lambda_1^3}{(\lambda_1+\mu)^2} q_1^4 + \frac{\lambda_2^3}{(\lambda_2+\mu)^2} q_2^4 \\
&\quad + \left(\frac{\lambda_1^3}{(\lambda_1+\mu)^2} + \frac{\lambda_2^3}{(\lambda_2+\mu)^2}\right) q_1^2 q_2^2\,.
\end{aligned}$$

Since $\lambda_1 \geq \lambda_2 \geq 0$, it is trivial to show that

$$\begin{aligned}
&\frac{\lambda_1^3}{(\lambda_1+\mu)^2} + \frac{\lambda_2^3}{(\lambda_2+\mu)^2} \\
&\geq \frac{\lambda_1^2\lambda_2}{(\lambda_1+\mu)^2} + \frac{\lambda_1\lambda_2^2}{(\lambda_2+\mu)^2}\,,
\end{aligned}$$

where $=$ holds when $\lambda_1 = \lambda_2$. Therefore

$$\begin{aligned}
&\left(\sum_{i=1}^{2} \frac{\lambda_i^3}{(\lambda_i+\mu)^2} q_i^2\right) \sum_{i=1}^{2} q_i^2 \\
&\geq \frac{\lambda_1^3}{(\lambda_1+\mu)^2} q_1^4 + \frac{\lambda_2^3}{(\lambda_2+\mu)^2} q_2^4 \\
&\quad + \left(\frac{\lambda_1^2\lambda_2}{(\lambda_1+\mu)^2} + \frac{\lambda_1\lambda_2^2}{(\lambda_2+\mu)^2}\right) q_1^2 q_2^2 \\
&= \left(\sum_{i=1}^{2} \frac{\lambda_i^2}{(\lambda_i+\mu)^2} q_i^2\right) \sum_{i=1}^{2} \lambda_i q_i^2\,,
\end{aligned}$$

so the property is true for $L = 2$, where $=$ holds when $\lambda_1 = \lambda_2$ or at least one of q_1 and q_2 is equal to 0.

- Inductive Step: Assume that the property is true for $L=m$, i. e.,

$$\left(\sum_{i=1}^{m}\frac{\lambda_i^3}{(\lambda_i+\mu)^2}q_i^2\right)\sum_{i=1}^{m}q_i^2 \geq \left(\sum_{i=1}^{m}\frac{\lambda_i^2}{(\lambda_i+\mu)^2}q_i^2\right)\sum_{i=1}^{m}\lambda_i q_i^2 \,. \tag{43.37}$$

We must prove that it is also true for $L=m+1$. As a matter of fact,

$$\begin{aligned}
&\left(\sum_{i=1}^{m+1}\frac{\lambda_i^3}{(\lambda_i+\mu)^2}q_i^2\right)\sum_{i=1}^{m+1}q_i^2 \\
&=\left(\sum_{i=1}^{m}\frac{\lambda_i^3}{(\lambda_i+\mu)^2}q_i^2+\frac{\lambda_{m+1}^3}{(\lambda_{m+1}+\mu)^2}q_{m+1}^2\right) \\
&\quad\times\left(\sum_{i=1}^{m}q_i^2+q_{m+1}^2\right) \\
&=\left(\sum_{i=1}^{m}\frac{\lambda_i^3}{(\lambda_i+\mu)^2}q_i^2\right)\sum_{i=1}^{m}q_i^2 \\
&\quad+\frac{\lambda_{m+1}^3}{(\lambda_{m+1}+\mu)^2}q_{m+1}^4 \\
&\quad+\sum_{i=1}^{m}\left(\frac{\lambda_i^3}{(\lambda_i+\mu)^2}+\frac{\lambda_{m+1}^3}{(\lambda_{m+1}+\mu)^2}\right)q_i^2q_{m+1}^2 \,.
\end{aligned} \tag{43.38}$$

Using the induction hypothesis, and also the fact that

$$\begin{aligned}
&\frac{\lambda_i^3}{(\lambda_i+\mu)^2}+\frac{\lambda_{m+1}^3}{(\lambda_{m+1}+\mu)^2} \\
&\geq \frac{\lambda_i^2\lambda_{m+1}}{(\lambda_i+\mu)^2}+\frac{\lambda_i\lambda_{m+1}^2}{(\lambda_{m+1}+\mu)^2} \,,
\end{aligned} \tag{43.39}$$

we obtain

$$\begin{aligned}
&\left(\sum_{i=1}^{m+1}\frac{\lambda_i^3}{(\lambda_i+\mu)^2}q_i^2\right)\sum_{i=1}^{m+1}q_i^2 \\
&\geq \left(\sum_{i=1}^{m}\frac{\lambda_i^2}{(\lambda_i+\mu)^2}q_i^2\right)\sum_{i=1}^{m}\lambda_i q_i^2 \\
&\quad+\frac{\lambda_{m+1}^3}{(\lambda_{m+1}+\mu)^2}q_{m+1}^4 \\
&\quad+\sum_{i=1}^{m}\left(\frac{\lambda_i^2\lambda_{m+1}}{(\lambda_i+\mu)^2}+\frac{\lambda_i\lambda_{m+1}^2}{(\lambda_{m+1}+\mu)^2}\right)q_i^2q_{m+1}^2 \\
&=\left(\sum_{i=1}^{m+1}\frac{\lambda_i^2}{(\lambda_i+\mu)^2}q_i^2\right)\sum_{i=1}^{m+1}\lambda_i q_i^2 \,,
\end{aligned} \tag{43.40}$$

where $=$ holds when all the λ_is corresponding to nonzero q_i are equal, where $i=1,2,\ldots,n+1$. That completes the proof.

From *Lemma 1*, if we set $\mu=1$ and $q_i=b_{i1}$, we obtain the following inequality (see also the Appendix in [43.18]):

$$\frac{\sum_{i=1}^{L}\frac{\lambda_i^3}{(1+\lambda_i)^2}b_{i1}^2}{\sum_{i=1}^{L}\frac{\lambda_i^2}{(1+\lambda_i)^2}b_{i1}^2} \geq \frac{\sum_{i=1}^{L}\lambda_i b_{i1}^2}{\sum_{i=1}^{L}b_{i1}^2} = \mathrm{SNR} \,, \tag{43.41}$$

with equality if and only if all the λ_is corresponding to nonzero b_{i1} are equal, where $i=1,2,\cdots,L$. It follows immediately that

$$\xi_{\mathrm{nr}}(\boldsymbol{h}_{\mathrm{o}}) \geq \frac{\mathrm{SNR}+2}{\mathrm{SNR}} \,. \tag{43.42}$$

The right-hand side of (43.42) is always greater than 1 since SNR is nonnegative. This shows that noise reduction is always feasible with the Wiener filter. It can be checked from (43.42) that the lower bound of the noise-reduction factor is a monotonically decreasing function on SNR. It approaches infinity when the SNR comes close to 0 and approaches 1 as the SNR approaches infinity. This indicates that more noise reduction can be achieved with the Wiener filter as the SNR decreases, which is of course desirable since, as the SNR drops, there will be more noise to eliminate.

The speech-distortion index due to the Wiener filter, according to (43.8), can be written as

$$\varphi_{\mathrm{sd}}(\boldsymbol{h}_{\mathrm{o}}) = \frac{(\boldsymbol{h}_1-\boldsymbol{h}_{\mathrm{o}})^{\mathrm{T}}\boldsymbol{R}_x(\boldsymbol{h}_1-\boldsymbol{h}_{\mathrm{o}})}{\boldsymbol{h}_1^{\mathrm{T}}\boldsymbol{R}_x\boldsymbol{h}_1} \,. \tag{43.43}$$

Obviously, we have

$$\varphi_{\mathrm{sd}}(\boldsymbol{h}_{\mathrm{o}}) \geq 0 \,. \tag{43.44}$$

Substituting (43.30c) into (43.43), we derive

$$\begin{aligned}
\varphi_{\mathrm{sd}}(\boldsymbol{h}_{\mathrm{o}}) &= \frac{\sum_{i=1}^{L}\frac{\lambda_i}{(1+\lambda_i)^2}b_{i1}^2}{\sum_{i=1}^{L}\lambda_i b_{i1}^2} \\
&\leq \frac{\sum_{i=1}^{L}\frac{\lambda_i}{1+2\lambda_i}b_{i1}^2}{\sum_{i=1}^{L}\frac{\lambda_i+2\lambda_i^2}{1+2\lambda_i}b_{i1}^2} \leq \frac{1}{2\mathrm{SNR}+1} \,,
\end{aligned} \tag{43.45}$$

where we have used the following inequality:

$$\frac{\sum_{i=1}^{L}\frac{\lambda_i^2}{1+2\lambda_i}b_{i1}^2}{\sum_{i=1}^{L}\frac{\lambda_i}{1+2\lambda_i}b_{i1}^2} \geq \frac{\sum_{i=1}^{L}\lambda_i b_{i1}^2}{\sum_{i=1}^{L}b_{i1}^2} = \mathrm{SNR}\,. \tag{43.46}$$

This inequality can be proved via induction (following the same analysis steps in the proof of *Lemma 1*), which is left to the reader's investigation. Figure 43.1 illustrates expressions (43.42) and (43.45).

From the previous analysis we see that, while noise reduction is feasible with the Wiener filter, speech attenuation is also unavoidable. In general, the more the noise is reduced, the more the speech is attenuated. A key question is whether the Wiener filter can improve SNR. To answer this question, we give the following proposition.

Proposition 1. With the Wiener filter given in (43.18) and (43.23), the a posteriori SNR (defined after the Wiener filter) is always greater than or at least equal to the a priori SNR.

Proof. If the noise $v(n)$ is zero, we already see that the Wiener filter has no effect on the speech signal. Now we consider the case where the noise is not zero. From (43.33), we know that the a priori SNR is

$$\mathrm{SNR} = \frac{\sum_{i=1}^{L}\lambda_i b_{i1}^2}{\sum_{i=1}^{L}b_{i1}^2}\,. \tag{43.47}$$

After the Wiener filter, the a posteriori SNR can be written as

$$\mathrm{SNR}_\mathrm{o} = \frac{\boldsymbol{h}_\mathrm{o}^\mathrm{T}\boldsymbol{R}_x\boldsymbol{h}_\mathrm{o}}{\boldsymbol{h}_\mathrm{o}^\mathrm{T}\boldsymbol{R}_v\boldsymbol{h}_\mathrm{o}}\,. \tag{43.48}$$

Substituting $\boldsymbol{h}_\mathrm{o} = \boldsymbol{R}_y^{-1}\boldsymbol{R}_x\boldsymbol{h}_1$ into (43.48), we obtain

$$\mathrm{SNR}_\mathrm{o} = \frac{\boldsymbol{h}_1^\mathrm{T}\boldsymbol{R}_x\boldsymbol{R}_y^{-1}\boldsymbol{R}_x\boldsymbol{R}_y^{-1}\boldsymbol{R}_x\boldsymbol{h}_1}{\boldsymbol{h}_1^\mathrm{T}\boldsymbol{R}_x\boldsymbol{R}_y^{-1}\boldsymbol{R}_v\boldsymbol{R}_y^{-1}\boldsymbol{R}_x\boldsymbol{h}_1}\,. \tag{43.49}$$

Using the matrix decomposition given in (43.30c), we deduce that

$$\mathrm{SNR}_\mathrm{o} = \frac{\sum_{i=1}^{L}\frac{\lambda_i^3}{(\lambda_i+1)^2}b_{i1}^2}{\sum_{i=1}^{L}\frac{\lambda_i^2}{(\lambda_i+1)^2}b_{i1}^2}\,. \tag{43.50}$$

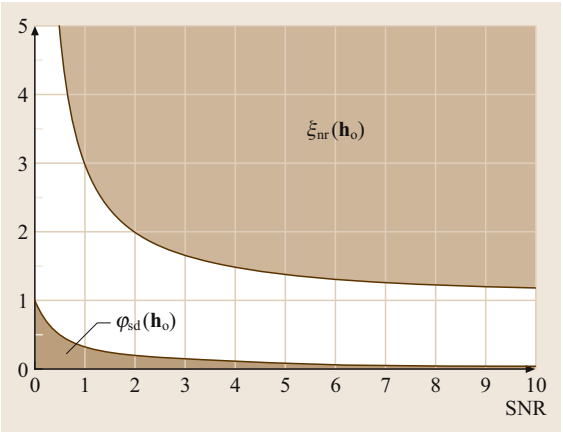


Fig. 43.1 Illustration of the areas where $\xi_{\mathrm{nr}}(\boldsymbol{h}_\mathrm{o})$ and $\varphi_{\mathrm{sd}}(\boldsymbol{h}_\mathrm{o})$ take their values as a function of the SNR

Following the inequality given in (43.41), we immediately have

$$\mathrm{SNR}_\mathrm{o} \geq \mathrm{SNR}\,. \tag{43.51}$$

This completes the proof of the proposition. Therefore, we see that the time-domain Wiener filter increases the SNR of the observed signal. In other words, the Wiener filter always reduces noise.

43.4.2 A Suboptimal Filter

From the previous analysis, we see that the time-domain Wiener filter achieves noise reduction by paying a price of speech distortion. A key question immediately arises: if a given application requires high-quality speech, is there any way that we can control the compromise between noise reduction and speech distortion? To answer this question, we describe a suboptimal filter.

The Wiener filter given in (43.23) has a nice physical interpretation: it is the sum of two filters, i. e., $\boldsymbol{h}_1$ and $-\boldsymbol{R}_y^{-1}\boldsymbol{r}_{vv}$, each of which serves a different purpose. The objective of the first is to create a replica of the noisy signal, while the aim of the second is to generate a noise estimate. So the Wiener filter actually achieves noise reduction in two steps: it first creates an optimal noise estimate, and then subtracts that estimate from the noisy observations. Obviously, if we introduce a parameter to control the amount of noise to be subtracted, we can manage the tradeoff between noise reduction and speech distortion. Therefore, we construct the following

filter:

$$\boldsymbol{h}_{\text{sub}} = \boldsymbol{h}_1 - \alpha \boldsymbol{R}_y^{-1} \boldsymbol{r}_{vv} \,, \tag{43.52}$$

where $\alpha \geq 0$ is a real number. Note that $\boldsymbol{h}_{\text{sub}}$ is no longer the solution of the MMSE criterion, so we shall call it a suboptimal filter.

Substituting $\boldsymbol{h}_{\text{sub}}$ into (43.13), we can derive the corresponding MSE due to the suboptimal filter, i. e.,

$$\begin{aligned} J_x(\boldsymbol{h}_{\text{sub}}) &= E\left\{\left[x(n) - \boldsymbol{h}_{\text{sub}}^{\text{T}} \boldsymbol{y}(n)\right]^2\right\} \\ &= \sigma_v^2 - \alpha(2-\alpha)\boldsymbol{r}_{vv}^{\text{T}} \boldsymbol{R}_y^{-1} \boldsymbol{r}_{vv} \,. \end{aligned} \tag{43.53}$$

In order to have noise reduction, α must be chosen in such a way that $J_x(\boldsymbol{h}_{\text{sub}}) < J_x(\boldsymbol{h}_1)$, therefore,

$$0 < \alpha < 2 \,. \tag{43.54}$$

The noise-reduction factor of the suboptimal filter can be written as

$$\xi_{\text{nr}}(\boldsymbol{h}_{\text{sub}}) = \frac{E\left\{\left[\boldsymbol{h}_1^{\text{T}} \boldsymbol{v}(n)\right]^2\right\}}{E\left\{\left[\boldsymbol{h}_{\text{sub}}^{\text{T}} \boldsymbol{v}(n)\right]^2\right\}} \,. \tag{43.55}$$

With the matrix decomposition given in (43.30c), we can further rewrite this factor in the form

$$\xi_{\text{nr}}(\boldsymbol{h}_{\text{sub}}) = \frac{\sum\limits_{i=1}^{L} b_{i1}^2}{\sum\limits_{i=1}^{L} \frac{(\lambda_i+1-\alpha)^2}{(1+\lambda_i)^2} b_{i1}^2} \,. \tag{43.56}$$

Similarly, we can express the speech-distortion index as

$$\begin{aligned} \varphi_{\text{sd}}(\boldsymbol{h}_{\text{sub}}) &= \frac{E\left\{\left[x(n) - \boldsymbol{h}_{\text{sub}}^{\text{T}} \boldsymbol{x}(n)\right]^2\right\}}{\sigma_x^2} \\ &= \alpha^2 \frac{\sum\limits_{i=1}^{L} \frac{\lambda_i}{(1+\lambda_i)^2} b_{i1}^2}{\sum\limits_{i=1}^{L} \lambda_i b_{i1}^2} = \alpha^2 \varphi_{\text{sd}}(\boldsymbol{h}_{\text{o}}) \,. \end{aligned} \tag{43.57}$$

So the ratio between the speech-distortion indices corresponding to the two filters $\boldsymbol{h}_{\text{sub}}$ and $\boldsymbol{h}_{\text{o}}$ depends on α only.

In order to have less distortion with the suboptimal filter $\boldsymbol{h}_{\text{sub}}$ than with the Wiener filter $\boldsymbol{h}_{\text{o}}$, we must find α in such a way that,

$$\varphi_{\text{sd}}(\boldsymbol{h}_{\text{sub}}) < \varphi_{\text{sd}}(\boldsymbol{h}_{\text{o}}) \,. \tag{43.58}$$

The condition on α should be $-1 < \alpha < 1$. Finally, if α is taken as $0 < \alpha < 1$, the suboptimal filter $\boldsymbol{h}_{\text{sub}}$ reduces the level of noise in the observed signal $y(n)$ while causing less distortion than the Wiener filter $\boldsymbol{h}_{\text{o}}$. For the extreme case $\alpha = 0$, we obtain $\boldsymbol{h}_{\text{sub}} = \boldsymbol{h}_1$, so there is no noise reduction but no speech distortion as well. At the other extreme $\alpha = 1$, we have $\boldsymbol{h}_{\text{sub}} = \boldsymbol{h}_{\text{o}}$, hence noise reduction is maximized and so is speech distortion.

The a posteriori SNR, after the suboptimal filter, can be written

$$\text{SNR}_{\text{sub}} = \frac{\boldsymbol{h}_{\text{sub}}^{\text{T}} \boldsymbol{R}_x \boldsymbol{h}_{\text{sub}}}{\boldsymbol{h}_{\text{sub}}^{\text{T}} \boldsymbol{R}_v \boldsymbol{h}_{\text{sub}}} = \frac{\sum\limits_{i=1}^{L} \frac{\lambda_i(\lambda_i+1-\alpha)^2}{(\lambda_i+1)^2} b_{i1}^2}{\sum\limits_{i=1}^{L} \frac{(\lambda_i+1-\alpha)^2}{(\lambda_i+1)^2} b_{i1}^2} \,. \tag{43.59}$$

As long as $\alpha \in (0, 1)$, it can be shown that

$$\text{SNR}_{\text{sub}} \geq \text{SNR} \,. \tag{43.60}$$

The proof of this can easily be obtained by following the proof of *Proposition 1*. Therefore, the suboptimal filter can also improve SNR, although less effectively than the Wiener filter.

43.4.3 Subspace Method

We have seen that the Wiener filter, derived from the MMSE criterion, can achieve the maximum amount of noise reduction (as measured by the definition of the noise-reduction factor), but it causes significant speech distortion at the same time. In the previous section, we discussed a suboptimal filter, which was constructed empirically from the Wiener filter. While it makes good sense for managing the compromise between noise reduction and speech distortion, the suboptimal filter has no optimality properties associated with it. We now investigate a class of techniques called subspace methods, which, like the Wiener filter, acquire a clean speech estimate by applying a linear transformation to a vector of noisy speech observations. Similar to the Wiener filter, the subspace approach is also formulated from the MMSE criterion with mathematical rigor; hence it is also an optimal estimator. However, it differs from the Wiener filter in that its linear transformation is derived from a constrained optimization problem while the Wiener filter is deduced from an unconstrained one.

Consider the signal model given in (43.2). An estimate of the clean speech can be obtained by applying a linear transformation to the noisy speech vector, i. e.,

$$\hat{\boldsymbol{x}}(n) = \boldsymbol{H}\boldsymbol{y}(n) \,, \tag{43.61}$$

where $\boldsymbol{H}$ is a matrix of size $L \times L$. The error signal obtained by this estimation is written as

$$\begin{aligned}\boldsymbol{e}(n) &\triangleq \hat{\boldsymbol{x}}(n) - \boldsymbol{x}(n) = \boldsymbol{H}\boldsymbol{y}(n) - \boldsymbol{x}(n) \\ &= (\boldsymbol{H} - \boldsymbol{I})\boldsymbol{x}(n) + \boldsymbol{H}\boldsymbol{v}(n) = \boldsymbol{e}_x(n) + \boldsymbol{e}_v(n) \,,\end{aligned} \tag{43.62}$$

where

$$\boldsymbol{e}_x(n) \triangleq (\boldsymbol{H} - \boldsymbol{I})\boldsymbol{x}(n) \tag{43.63}$$

and

$$\boldsymbol{e}_v(n) \triangleq \boldsymbol{H}\boldsymbol{v}(n) \tag{43.64}$$

represent, respectively, the speech distortion due to the linear transformation and the residual noise. It is immediately seen that there are three criteria to estimate $\boldsymbol{H}$:

1. minimizing the energy of $\boldsymbol{e}(n)$;
2. minimizing the energy of $\boldsymbol{e}_v(n)$ while limiting the level of speech distortion;
3. minimizing the energy of $\boldsymbol{e}_x(n)$ while limiting the level of residual noise.

The first case will lead to the Wiener solution. The only difference between the Wiener filter discussed in Sect. 43.4.1 and this estimator is that the former obtains the current speech estimate using only past and current noisy speech samples, while the latter uses not only past and present observation samples, but also future values. The second criterion is rarely used in practice because the resulting estimator produces nonstationary residual noise (due to the nonstationarity of speech signals), which is usually intolerable to the human perception system. The third case will lead to the so-called subspace method, which, like the Wiener filter, has been widely investigated.

Mathematically, the optimal linear transformation in the subspace technique can be described as

$$\begin{aligned}&\boldsymbol{H}_{\mathrm{o}} = \arg\min_{\boldsymbol{H}} \mathrm{tr}\left\{E\left[\boldsymbol{e}_x(n)\boldsymbol{e}_x^{\mathrm{T}}(n)\right]\right\} \\ &\text{subject to } \mathrm{tr}\left\{E\left[\boldsymbol{e}_v(n)\boldsymbol{e}_v^{\mathrm{T}}(n)\right]\right\} \le L\sigma^2 \,.\end{aligned} \tag{43.65}$$

If we use a Lagrange multiplier to adjoin the constraint to the cost function, (43.65) can be rewritten as

$$\boldsymbol{H}_{\mathrm{o}} = \arg\min_{\boldsymbol{H}} \mathcal{L}(\boldsymbol{H}, \mu) \,, \tag{43.66}$$

where

$$\begin{aligned}&\mathcal{L}(\boldsymbol{H}, \mu) \\ &= \mathrm{tr}\left\{E\left[\boldsymbol{e}_x(n)\boldsymbol{e}_x^{\mathrm{T}}(n)\right]\right\} \\ &\quad + \mu\left(\mathrm{tr}\left\{E\left[\boldsymbol{e}_v(n)\boldsymbol{e}_v^{\mathrm{T}}(n)\right]\right\} - L\sigma^2\right) \\ &= \mathrm{tr}\left[(\boldsymbol{H} - \boldsymbol{I})\boldsymbol{R}_x(\boldsymbol{H} - \boldsymbol{I})^{\mathrm{T}}\right] \\ &\quad + \mu\left[\mathrm{tr}\left(\boldsymbol{H}\boldsymbol{R}_v\boldsymbol{H}^{\mathrm{T}}\right) - L\sigma^2\right] \,,\end{aligned} \tag{43.67}$$

$\boldsymbol{R}_x$ and $\boldsymbol{R}_v$ are, respectively, the covariance matrices of the clean speech and noise, and $\mu > 0$ is the Lagrange multiplier. Using the fact that

$$\frac{\partial}{\partial \boldsymbol{H}}\mathrm{tr}(\boldsymbol{R}_x\boldsymbol{H}) = \frac{\partial}{\partial \boldsymbol{H}}\mathrm{tr}(\boldsymbol{H}\boldsymbol{R}_x) = \boldsymbol{R}_x^{\mathrm{T}} = \boldsymbol{R}_x \,, \tag{43.68a}$$

$$\frac{\partial}{\partial \boldsymbol{H}}\mathrm{tr}\left(\boldsymbol{H}\boldsymbol{R}_x\boldsymbol{H}^{\mathrm{T}}\right) = 2\boldsymbol{H}\boldsymbol{R}_x \,, \tag{43.68b}$$

$$\frac{\partial}{\partial \boldsymbol{H}}\mathrm{tr}\left(\boldsymbol{H}\boldsymbol{R}_v\boldsymbol{H}^{\mathrm{T}}\right) = 2\boldsymbol{H}\boldsymbol{R}_v \,, \tag{43.68c}$$

we can readily deduce that

$$\frac{\partial}{\partial \boldsymbol{H}}\mathcal{L}(\boldsymbol{H}, \mu) = 2\boldsymbol{H}\boldsymbol{R}_x - 2\boldsymbol{R}_x + 2\mu\boldsymbol{H}\boldsymbol{R}_v \,. \tag{43.69}$$

Equating the right-hand side of (43.69) to zero, we obtain the solution to (43.66):

$$\boldsymbol{H}_{\mathrm{o}} = \boldsymbol{R}_x(\boldsymbol{R}_x + \mu\boldsymbol{R}_v)^{-1} \,, \tag{43.70}$$

where μ satisfies

$$\mathrm{tr}\left[\boldsymbol{R}_x(\boldsymbol{R}_x + \mu\boldsymbol{R}_v)^{-1}\boldsymbol{R}_v(\boldsymbol{R}_x + \mu\boldsymbol{R}_v)^{-1}\boldsymbol{R}_x\right] = L\sigma^2 \,. \tag{43.71}$$

To implement the optimal transformation given in (43.70), we need to compute the inverse of the matrix sum $\boldsymbol{R}_x + \mu\boldsymbol{R}_v$. The most straightforward way to do this is the direct method, which computes the inverse of the matrix explicitly, but this approach is not very robust particularly when the matrix is not well conditioned. Another popular method is based on either eigenvalue decomposition [43.19–25] or generalized eigenvalue decomposition [43.26]. For example, with the generalized eigenvalue decomposition as defined in (43.30c), we can rewrite the optimal transformation as

$$\boldsymbol{H}_{\mathrm{o}} = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{\Lambda}(\boldsymbol{\Lambda} + \mu\boldsymbol{I})^{-1}\boldsymbol{B}^{-\mathrm{T}} \,. \tag{43.72}$$

Therefore, the estimation of the clean speech can be decoupled into three steps: applying the transformation $\boldsymbol{B}^{-\mathrm{T}}$ to the noisy signal, modifying the noisy signal in the transformed domain by a gain function $\boldsymbol{\Lambda}(\boldsymbol{\Lambda} + \mu\boldsymbol{I})^{-1}$, and applying $\boldsymbol{B}^{\mathrm{T}}$ to transform the modified components back to the original domain.

Another interesting interpretation of (43.72) is to divide the vector space into two subspaces: the speech subspace corresponding to all the λ_i for $\lambda_i > 0$ and the noise subspace associated with all λ_i for $\lambda_i = 0$. Suppose that the dimension of the speech subspace is M. We then can rewrite (43.72) as

$$\boldsymbol{H}_{\mathrm{o}} = \boldsymbol{B}^{\mathrm{T}}\begin{pmatrix}\boldsymbol{\Sigma} & \boldsymbol{0}_{M\times K} \\ \boldsymbol{0}_{K\times M} & \boldsymbol{0}_{K\times K}\end{pmatrix}\boldsymbol{B}^{-\mathrm{T}} \,, \tag{43.73}$$

where $K = L - M$ is the order of the noise subspace and

$$\boldsymbol{\Sigma} = \text{diag}\left(\frac{\lambda_1}{\lambda_1+\mu}, \frac{\lambda_2}{\lambda_2+\mu}, \cdots, \frac{\lambda_M}{\lambda_M+\mu}\right) \tag{43.74}$$

is an $M \times M$ diagonal matrix. We now clearly see that noise reduction with the subspace method is achieved by nulling the noise subspace and cleaning the speech-plus-noise subspace via a reweighted reconstruction.

Now we are ready to check if the linear transformation given in (43.70) can really reduce the level of noise. Let us examine the noise-reduction factor, which can be written, according to (43.6), as

$$\xi_{\text{nr}}(\boldsymbol{H}_{\text{o}}) = \frac{\text{tr}\{\boldsymbol{R}_v\}}{\text{tr}\left\{E\left[\boldsymbol{e}_v(n)\boldsymbol{e}_v^{\text{T}}(n)\right]\right\}} . \tag{43.75}$$

Substituting (43.70) into (43.64), we can derive

$$\begin{aligned}&\text{tr}\left\{E\left[\boldsymbol{e}_v(n)\boldsymbol{e}_v^{\text{T}}(n)\right]\right\}\\ &= \text{tr}\left\{\boldsymbol{R}_x(\boldsymbol{R}_x+\mu\boldsymbol{R}_v)^{-1}\boldsymbol{R}_v(\boldsymbol{R}_x+\mu\boldsymbol{R}_v)^{-1}\boldsymbol{R}_x\right\} .\end{aligned} \tag{43.76}$$

Following the matrix-decomposition procedure given in (43.30c), we easily deduce that

$$\text{tr}\{\boldsymbol{R}_v\} = \text{tr}\{\boldsymbol{B}^{\text{T}}\boldsymbol{B}\} = \text{tr}\{\boldsymbol{B}\boldsymbol{B}^{\text{T}}\} = \sum_{i=1}^{L}\sum_{j=1}^{L} b_{ij}^2 , \tag{43.77}$$

and

$$\begin{aligned}\text{tr}\left\{E\left[\boldsymbol{e}_v(n)\boldsymbol{e}_v^{\text{T}}(n)\right]\right\} &= \text{tr}\{\boldsymbol{B}^{\text{T}}\boldsymbol{\Lambda}(\boldsymbol{\Lambda}+\mu\boldsymbol{I})^{-2}\boldsymbol{\Lambda}\boldsymbol{B}\}\\ &= \text{tr}\{\boldsymbol{\Lambda}(\boldsymbol{\Lambda}+\mu\boldsymbol{I})^{-2}\boldsymbol{\Lambda}\boldsymbol{B}\boldsymbol{B}^{\text{T}}\}\\ &= \sum_{i=1}^{L}\sum_{j=1}^{L}\frac{\lambda_i^2}{(\lambda_i+\mu)^2}b_{ij}^2 .\end{aligned} \tag{43.78}$$

Therefore, we have

$$\xi_{\text{nr}}(\boldsymbol{H}_{\text{o}}) = \frac{\sum_{i=1}^{L}\sum_{j=1}^{L} b_{ij}^2}{\sum_{i=1}^{L}\sum_{j=1}^{L}\frac{\lambda_i^2}{(\lambda_i+\mu)^2}b_{ij}^2} . \tag{43.79}$$

Since $\mu > 0$ and $\lambda_i \geq 0$ for $i = 1, 2, \cdots, L$, it is obvious that

$$\xi_{\text{nr}}(\boldsymbol{H}_{\text{o}}) > 1 . \tag{43.80}$$

This shows that noise reduction is always feasible with the subspace method.

Since the energy of the speech distortion $\boldsymbol{e}_x(n)$ is always greater than zero, the power of the clean speech is also attenuated when we reduce the noise with the subspace method. We then ask the same question about SNR as we did for the Wiener filter: can the subspace method improve SNR? To answer this question, we give the following proposition.

Proposition 2. For the linear transformation given in (43.70), if $\mu > 0$, the a posteriori SNR with the subspace method is always greater than or equal to the a priori SNR.

Proof. If the noise is zero, we can easily check that the optimal transformation matrix $\boldsymbol{H}_{\text{o}}$ will be the identity matrix, and hence will not change the input speech. If the noise is not zero, the a priori SNR can be written as

$$\text{SNR} = \frac{\text{tr}\{\boldsymbol{R}_x\}}{\text{tr}\{\boldsymbol{R}_v\}} . \tag{43.81}$$

After applying the linear transformation $\boldsymbol{H}_{\text{o}}$, the a posteriori SNR can be expressed as

$$\begin{aligned}\text{SNR}_{\text{o}} &= \frac{\text{tr}\left\{E\left[\boldsymbol{H}_{\text{o}}\boldsymbol{x}(n)\boldsymbol{x}^{\text{T}}(n)\boldsymbol{H}_{\text{o}}^{\text{T}}\right]\right\}}{\text{tr}\left\{E\left[\boldsymbol{H}_{\text{o}}\boldsymbol{v}(n)\boldsymbol{v}^{\text{T}}(n)\boldsymbol{H}_{\text{o}}^{\text{T}}\right]\right\}}\\ &= \frac{\text{tr}\left\{\boldsymbol{H}_{\text{o}}\boldsymbol{R}_x\boldsymbol{H}_{\text{o}}^{\text{T}}\right\}}{\text{tr}\left\{\boldsymbol{H}_{\text{o}}\boldsymbol{R}_v\boldsymbol{H}_{\text{o}}^{\text{T}}\right\}} .\end{aligned} \tag{43.82}$$

It follows then that

$$\begin{aligned}&\frac{\text{SNR}_{\text{o}}}{\text{SNR}}\\ &= \frac{\text{tr}\{\boldsymbol{R}_v\}\text{tr}\left\{\boldsymbol{H}_{\text{o}}\boldsymbol{R}_x\boldsymbol{H}_{\text{o}}^{\text{T}}\right\}}{\text{tr}\{\boldsymbol{R}_x\}\text{tr}\left\{\boldsymbol{H}_{\text{o}}\boldsymbol{R}_v\boldsymbol{H}_{\text{o}}^{\text{T}}\right\}}\\ &= \frac{\text{tr}\{\boldsymbol{R}_v\}\text{tr}\left\{\boldsymbol{R}_x[\boldsymbol{R}_x+\mu\boldsymbol{R}_v]^{-1}\boldsymbol{R}_x[\boldsymbol{R}_x+\mu\boldsymbol{R}_v]^{-1}\boldsymbol{R}_x\right\}}{\text{tr}\{\boldsymbol{R}_x\}\text{tr}\left\{\boldsymbol{R}_x[\boldsymbol{R}_x+\mu\boldsymbol{R}_v]^{-1}\boldsymbol{R}_v[\boldsymbol{R}_x+\mu\boldsymbol{R}_v]^{-1}\boldsymbol{R}_x\right\}} .\end{aligned} \tag{43.83}$$

Following the matrix-decomposition procedure given in (43.30c) and using some simple algebra, we find that

$$\begin{aligned}&\frac{\text{SNR}_{\text{o}}}{\text{SNR}}\\ &= \frac{\text{tr}\{\boldsymbol{B}^{\text{T}}\boldsymbol{B}\}\text{tr}\{\boldsymbol{B}^{\text{T}}\boldsymbol{\Lambda}(\boldsymbol{\Lambda}+\mu\boldsymbol{I})^{-1}\boldsymbol{\Lambda}(\boldsymbol{\Lambda}+\mu\boldsymbol{I})^{-1}\boldsymbol{\Lambda}\boldsymbol{B}\}}{\text{tr}\{\boldsymbol{B}^{\text{T}}\boldsymbol{\Lambda}\boldsymbol{B}\}\text{tr}\{\boldsymbol{B}^{\text{T}}\boldsymbol{\Lambda}(\boldsymbol{\Lambda}+\mu\boldsymbol{I})^{-2}\boldsymbol{\Lambda}\boldsymbol{B}\}}\\ &= \frac{\text{tr}\{\boldsymbol{B}\boldsymbol{B}^{\text{T}}\}\text{tr}\{\boldsymbol{\Lambda}(\boldsymbol{\Lambda}+\mu\boldsymbol{I})^{-1}\boldsymbol{\Lambda}(\boldsymbol{\Lambda}+\mu\boldsymbol{I})^{-1}\boldsymbol{\Lambda}\boldsymbol{B}\boldsymbol{B}^{\text{T}}\}}{\text{tr}\{\boldsymbol{\Lambda}\boldsymbol{B}\boldsymbol{B}^{\text{T}}\}\text{tr}\{\boldsymbol{\Lambda}(\boldsymbol{\Lambda}+\mu\boldsymbol{I})^{-2}\boldsymbol{\Lambda}\boldsymbol{B}\boldsymbol{B}^{\text{T}}\}}\\ &= \frac{\sum_{i=1}^{L}\sum_{j=1}^{L} b_{ij}^2 \cdot \sum_{i=1}^{L}\sum_{j=1}^{L}\frac{\lambda_i^3}{(\lambda_i+\mu)^2}b_{ij}^2}{\sum_{i=1}^{L}\sum_{j=1}^{L}\lambda_i b_{ij}^2 \cdot \sum_{i=1}^{L}\sum_{j=1}^{L}\frac{\lambda_i^2}{(\lambda_i+\mu)^2}b_{ij}^2} .\end{aligned} \tag{43.84}$$

Now using the *Lemma 1* and setting $q_i^2 = \sum_{j=1}^{L} b_{ij}^2$, we obtain that

$$\sum_{i=1}^{L}\sum_{j=1}^{L} b_{ij}^2 \cdot \sum_{i=1}^{L}\sum_{j=1}^{L} \frac{\lambda_i^3}{(\lambda_i+\mu)^2} b_{ij}^2 \geq \sum_{i=1}^{L}\sum_{j=1}^{L} \lambda_i b_{ij}^2 \cdot \sum_{i=1}^{L}\sum_{j=1}^{L} \frac{\lambda_i^2}{(\lambda_i+\mu)^2} b_{ij}^2 \,. \tag{43.85}$$

It follows immediately that

$$\mathrm{SNR_o} \geq \mathrm{SNR} \,, \tag{43.86}$$

with equality if and only if $\lambda_1 = \lambda_2 = \cdots = \lambda_L$. Note that this condition is the same as that obtained previously for the Wiener filter, which should not come as a surprise since both techniques are formulated in a similar way.

From the previous analysis, we see that $\mathrm{SNR_o}$ depends not only on the speech and noise characteristics, but also on the value of μ. With some algebra, it can be shown that

$$\lim_{\mu\to 0} \mathrm{SNR_o} = \lim_{\mu\to 0} \frac{\sum_{i=1}^{L}\sum_{j=1}^{L} \frac{\lambda_i^3}{(\lambda_i+\mu)^2} b_{ij}^2}{\sum_{i=1}^{L}\sum_{j=1}^{L} \frac{\lambda_i^2}{(\lambda_i+\mu)^2} b_{ij}^2} = \mathrm{SNR} \,, \tag{43.87}$$

which is the lower bound of the a posteriori SNR. When $\mu \to +\infty$, we see that $\lambda_i + \mu \to \mu$. Therefore,

$$\lim_{\mu\to +\infty} \mathrm{SNR_o} = \frac{\sum_{i=1}^{L}\sum_{j=1}^{L} \lambda_i^3 b_{ij}^2}{\sum_{i=1}^{L}\sum_{j=1}^{L} \lambda_i^2 b_{ij}^2} \leq \sum_{i=1}^{L} \lambda_i \,, \tag{43.88}$$

which is the upper bound of the a posteriori SNR.

43.4.4 Frequency-Domain Wiener Filter

The Wiener filter can also be formulated in the frequency domain. One way of deriving such a filter is to transform the time-domain Wiener filter into the frequency domain using the so-called overlap-add technique. In this case, the time-domain Wiener filter and its frequency-domain counterpart have exactly the same performance. More often, however, the frequency-domain Wiener filter is formulated by directly estimating the clean speech spectrum from the noisy speech spectrum. The resulting filter differs in two aspects from the time-domain Wiener filter: first, the former is a causal filter, while the latter can be noncausal; second, the former is a full-band technique, while the latter is a subband technique, where each subband filter is independent of the filters corresponding to other frequency bands.

Let us consider the signal model in (43.3). The frequency-domain subband Wiener filter is derived by the criterion [43.46]

$$H_\mathrm{o}(\mathrm{i}\omega_k) = \arg \min_{H(\mathrm{i}\omega_k)} J_X[H(\mathrm{i}\omega_k)] \,, \tag{43.89}$$

where

$$J_X[H(\mathrm{i}\omega_k)] = E\left[|X(n,\mathrm{i}\omega_k) - H(\mathrm{i}\omega_k)Y(n,\mathrm{i}\omega_k)|^2\right]$$

is the MSE between the speech spectrum and its estimate at frequency ω_k. Differentiating $J_X[H(\mathrm{i}\omega_k)]$ with respect to $H(\mathrm{i}\omega_k)$ and equating the result to zero, we easily deduce the Wiener filter,

$$H_\mathrm{o}(\mathrm{i}\omega_k) = \frac{E\left[|X(n,\mathrm{i}\omega_k)|^2\right]}{E\left[|Y(n,\mathrm{i}\omega_k)|^2\right]} = \frac{P_x(\omega_k)}{P_y(\omega_k)} \,, \tag{43.90}$$

where

$$P_x(\omega_k) = \frac{1}{L} E\left[|X(n,\mathrm{i}\omega_k)|^2\right]$$

and

$$P_y(\omega_k) = \frac{1}{L} E\left[|Y(n,\mathrm{i}\omega_k)|^2\right]$$

are the power spectral densities (PSDs) of $x(n)$ and $y(n)$, respectively. It can be seen from this expression that the frequency-domain Wiener filter $H_\mathrm{o}(\mathrm{i}\omega_k)$ is nonnegative and real valued. Therefore, it only modifies the amplitude of the noisy speech spectra, while leaving the phase components unchanged. Since $H_\mathrm{o}(\mathrm{i}\omega_k)$ is real valued, we shall, from now on, drop the symbol i from its expression, which should not cause any confusion.

We see from (43.90) that, in order to obtain the Wiener filter, we need to know the PSDs of both the noisy and clean speech signals. The former can be directly estimated from the noisy observation signal $y(n)$. But $x(n)$ is not accessible; hence it may seem difficult to estimate the PSD of the clean speech. However, since speech and noise are assumed to be uncorrelated, we have

$$P_y(\omega_k) = P_x(\omega_k) + P_v(\omega_k) \,, \tag{43.91}$$

where $P_v(\omega_k)$ is the PSD of $v(n)$. Therefore, the Wiener filter can be written as

$$H_\mathrm{o}(\omega_k) = \frac{P_y(\omega_k) - P_v(\omega_k)}{P_y(\omega_k)} \,. \tag{43.92}$$

Now we see the Wiener filter depends on the PSDs of both the noisy observation and the noise signal, where the noise PSD can be estimated during the absence of speech.

The optimal estimate of the clean speech spectrum, using $H_o(\omega_k)$, is

$$\begin{aligned}\hat{X}_o(n, i\omega_k) &= H_o(\omega_k)Y(n, i\omega_k) \\ &= H_o(\omega_k)X(n, i\omega_k) + H_o(\omega_k)V(n, i\omega_k) .\end{aligned} \tag{43.93}$$

Applying the inverse DFT (IDFT) to (43.93), we can obtain the optimal estimate of the speech samples $\hat{x}_o(n)$.

The power of the estimated clean speech can be evaluated according to Parseval's relation, i. e.,

$$\begin{aligned}&E\left[\hat{x}_o^2(n)\right] \\ &= \sum_{k=0}^{L-1} \frac{1}{L} E\left[|\hat{X}_o(n, i\omega_k)|^2\right] \\ &= \sum_{k=0}^{L-1} H_o^2(\omega_k) P_y(\omega_k) \\ &= \sum_{k=0}^{L-1} \frac{P_x^2(\omega_k)}{P_y^2(\omega_k)} P_x(\omega_k) + \sum_{k=0}^{L-1} \frac{P_x^2(\omega_k)}{P_y^2(\omega_k)} P_v(\omega_k) ,\end{aligned} \tag{43.94}$$

which is the sum of two terms. The first is the power of the filtered clean speech and the second is the power of the residual noise.

If the noise is not zero, we can write the noise-reduction factor of the frequency-domain Wiener filter, according to (43.6) and (43.94), as

$$\xi_{nr}[H(\omega_k)] = \frac{\sum_{k=0}^{L-1} P_v(\omega_k)}{\sum_{k=0}^{L-1} \frac{P_x^2(\omega_k)}{P_y^2(\omega_k)} P_v(\omega_k)} . \tag{43.95}$$

Since $P_x^2(\omega_k) \le P_y^2(\omega_k)$, we can easily verify that

$$\xi_{nr}[H(\omega_k)] \ge 1 . \tag{43.96}$$

This indicates that the Wiener filter can reduce the noise level (unless there is no noise). Similarly, we can check that the power of the filtered clean speech is always less than the power of the original clean speech. So, the frequency-domain Wiener filter, just like its time-domain counterpart, also reduces noise at the price of attenuating the clean speech. It is key to know, then, whether this Wiener filter can improve SNR. We have the following proposition:

Proposition 3. With the Wiener filter given in (43.90), the a posteriori SNR (defined after the Wiener filter) is always greater than or equal to the a priori SNR.

Proof. If there is no noise, we see that the Wiener filter has no effect on SNR. Now we consider the generic case where noise is not zero. In the frequency domain, the a priori SNR can be expressed as

$$\text{SNR} = \frac{\sum_{k=0}^{L-1} P_x(\omega_k)}{\sum_{k=0}^{L-1} P_v(\omega_k)} . \tag{43.97}$$

Similarly, we can write, after Wiener filtering, the a posteriori SNR according to (43.94) as

$$\text{SNR}_o = \frac{\sum_{k=0}^{L-1} \frac{P_x^2(\omega_k)}{P_y^2(\omega_k)} P_x(\omega_k)}{\sum_{k=0}^{L-1} \frac{P_x^2(\omega_k)}{P_y^2(\omega_k)} P_v(\omega_k)} . \tag{43.98}$$

Now let us denote

$$\begin{aligned}\phi(\omega_k) = &\sum_{k=0}^{L-1} \frac{P_x^2(\omega_k)}{P_y^2(\omega_k)} P_x(\omega_k) \sum_{k=0}^{L-1} P_v(\omega_k) \\ &- \sum_{k=0}^{L-1} \frac{P_x^2(\omega_k)}{P_y^2(\omega_k)} P_v(\omega_k) \sum_{k=0}^{L-1} P_x(\omega_k) .\end{aligned}$$

It follows immediately that

$$\begin{aligned}&\phi(\omega_k) \\ &= \sum_{k=0}^{L-1} \sum_{j=0}^{L-1} \frac{P_x^3(\omega_k)}{P_y^2(\omega_k)} P_v(\omega_j) \\ &\quad - \sum_{k=0}^{L-1} \sum_{j=0}^{L-1} \frac{P_x^2(\omega_k)}{P_y^2(\omega_k)} P_v(\omega_k) P_x(\omega_j) \\ &= \sum_{k=0}^{L-1} \sum_{j=0}^{L-1} \frac{P_x^2(\omega_k)}{P_y^2(\omega_k)} [P_x(\omega_k)P_v(\omega_j) - P_v(\omega_k)P_x(\omega_j)] \\ &= \sum_{k=0}^{L-1} \sum_{j>k}^{L-1} \left[\frac{P_x^2(\omega_k)}{P_y^2(\omega_k)} - \frac{P_x^2(\omega_j)}{P_y^2(\omega_j)}\right] \\ &\quad \times [P_x(\omega_k)P_v(\omega_j) - P_v(\omega_k)P_x(\omega_j)] .\end{aligned} \tag{43.99}$$

Using (43.91), we can rewrite $\phi(\omega_k)$ as

$$\begin{aligned}&\phi(\omega_k)\\&=\sum_{k=0}^{L-1}\sum_{j>k}^{L-1}\frac{1}{P_y(\omega_k)P_y(\omega_j)}\left[\frac{P_x(\omega_k)}{P_y(\omega_k)}+\frac{P_x(\omega_j)}{P_y(\omega_j)}\right]\\&\quad\times[P_x(\omega_k)P_v(\omega_j)-P_v(\omega_k)P_x(\omega_j)]^2\,.\end{aligned}\tag{43.100}$$

Since $P_x(\omega_k)\geq 0$, $P_v(\omega_k)\geq 0$, and $P_y(\omega_k)\geq 0$, it is easy to see that $\phi(\omega_k)$ is greater than, or at least equal to 0. Therefore, we have

$$\begin{aligned}&\sum_{k=0}^{L-1}\frac{P_x^2(\omega_k)}{P_y^2(\omega_k)}P_x(\omega_k)\sum_{k=0}^{L-1}P_v(\omega_k)\\&\geq\sum_{k=0}^{L-1}\frac{P_x^2(\omega_k)}{P_y^2(\omega_k)}P_v(\omega_k)\sum_{k=0}^{L-1}P_x(\omega_k)\,,\end{aligned}$$

which means that

$$\mathrm{SNR_o}\geq\mathrm{SNR}\,,$$

where we see from (43.100) that equality is attained if and only if $P_x(\omega_0)/P_v(\omega_0)=P_x(\omega_1)/P_v(\omega_1)=\cdots=P_x(\omega_{L-1})/P_v(\omega_{L-1})$. In other words, the frequency-domain Wiener filter will increase SNR unless all subband SNRs are equal, which is understandable. When all subband SNRs are equal, both speech and noise have the same PSD under the given resolution condition. In this case, the Wiener filter is not able to distinguish noise from speech, and hence is not able to increase SNR. In all other cases, the subband Wiener filter will improve SNR.

43.4.5 Short-Time Parametric Wiener Filter

In implementations of the Wiener filter [(43.90) or (43.92)], some approximations are required since the PSDs of the noisy speech and noise signals are not known and must be estimated. One way is to replace the PSD with the instantaneous magnitude-squared spectral amplitude, yielding

$$H_{\mathrm{MS}}(\omega_k)=\frac{|Y(n,\mathrm{i}\omega_k)|^2-|V(n,\mathrm{i}\omega_k)|^2}{|Y(n,\mathrm{i}\omega_k)|^2}\,.\tag{43.101}$$

With such a filter, the clean speech estimate can be written as

$$\hat{X}(n,\mathrm{i}\omega_k)=H_{\mathrm{MS}}(\omega_k)Y(\mathrm{i}\omega_k)\,.\tag{43.102}$$

To enable more flexibility in manipulating the trade-off between noise suppression and speech distortion, (43.101) has been extended to a more-generic form referred to as the parametric Wiener filtering technique [43.13–15], namely

$$H_{\mathrm{PW}}(\omega_k)=\left[\frac{|Y(n,\mathrm{i}\omega_k)|^p-\eta|V(n,\mathrm{i}\omega_k)|^p}{|Y(n,\mathrm{i}\omega_k)|^p}\right]^{1/q}\,,\tag{43.103}$$

where p and q are some positive numbers, and η is a parameter introduced to control the amount of noise to be reduced. More-aggressive noise reduction can be achieved with $\eta>1$; this is, of course, at the expense of distortion artifacts. If we are more concerned with speech distortion, we can choose $\eta<1$. Commonly used configurations of (p,q,η) include $(1,1,1)$, $(2,1,1)$, and $(2,1/2,1)$.

With the short-time parametric Wiener filter, the estimate of the clean speech spectrum is written as

$$\hat{X}(n,\mathrm{i}\omega_k)=H_{\mathrm{PW}}(\omega_k)Y(n,\mathrm{i}\omega_k)\,.\tag{43.104}$$

Note that for the general configuration of (p,q,η), the estimator in (43.104) has no optimality properties associated with it. However, this does not diminish its usefulness. As a matter of fact, (43.104) has been widely used as a benchmark to which other noise reduction techniques are compared [43.13, 14, 27, 29].

The analysis of noise reduction, speech distortion, and SNR of the parametric Wiener filter can be done by following the analysis for the frequency-domain Wiener filter. This will be left to the reader's investigation.

43.5 Noise Reduction via Spectral Restoration

In the frequency domain, noise reduction can be formulated as a robust spectral estimation problem, i. e., estimation of the clean speech spectrum from the noisy speech spectrum. We now develop good estimators of clean speech within the framework of estimation theory.

43.5.1 MMSE Spectral Estimator

The minimum-mean-square-error (MMSE) estimator has its roots in Bayesian estimation theory. Let us consider the signal model given in (43.3). Since both clean speech and noise are assumed to be zero-mean random

processes, their short-time Fourier transform (STFT) coefficients, i. e., $X(n, \mathrm{i}\omega_k)$ and $V(n, \mathrm{i}\omega_k)$, are also zero-mean random variables. If we decompose the clean speech spectrum into real and imaginary parts as

$$X(n, \mathrm{i}\omega_k) = X_\mathrm{R} + \mathrm{i}X_\mathrm{I} ,$$

and assume that the two random variables X_R and X_I are independent, then an MMSE (conditional) estimator of $X(n, \mathrm{i}\omega_k)$ is written as

$$\begin{aligned}
&\widehat{X}_{\mathrm{MMSE}}(n, \mathrm{i}\omega_k) \\
&= E\left[X(n, \mathrm{i}\omega_k)|Y(n, \mathrm{i}\omega_k)\right] \\
&= E\left[X_\mathrm{R}|Y(n, \mathrm{i}\omega_k)\right] + \mathrm{i}E\left[X_\mathrm{I}|Y(n, \mathrm{i}\omega_k)\right] .
\end{aligned} \tag{43.105}$$

Let us define

$$\begin{aligned}
\widehat{X}_{\mathrm{R,MMSE}} &\triangleq E\left[X_\mathrm{R}|Y(n, \mathrm{i}\omega_k)\right] , \\
\widehat{X}_{\mathrm{I,MMSE}} &\triangleq E\left[X_\mathrm{I}|Y(n, \mathrm{i}\omega_k)\right] .
\end{aligned} \tag{43.106}$$

Using the conditional and joint probability density functions (pdfs), we can write the two estimators in (43.106) as

$$\begin{aligned}
&\widehat{X}_{\mathrm{R,MMSE}} \\
&= \int X_\mathrm{R} p[X_\mathrm{R}|Y(n, \mathrm{i}\omega_k)]\,\mathrm{d}X_\mathrm{R} \\
&= \frac{\iint X_\mathrm{R} p[Y(n, \mathrm{i}\omega_k)|X_\mathrm{R}, X_\mathrm{I}]p(X_\mathrm{R}, X_\mathrm{I})\,\mathrm{d}X_\mathrm{R}\,\mathrm{d}X_\mathrm{I}}{\iint p[Y(n, \mathrm{i}\omega_k)|X_\mathrm{R}, X_\mathrm{I}]p(X_\mathrm{R}, X_\mathrm{I})\,\mathrm{d}X_\mathrm{R}\,\mathrm{d}X_\mathrm{I}} ,
\end{aligned} \tag{43.107}$$

and

$$\begin{aligned}
&\widehat{X}_{\mathrm{I,MMSE}} \\
&= \int X_\mathrm{I} p[X_\mathrm{I}|Y(n, \mathrm{i}\omega_k)]\,\mathrm{d}X_\mathrm{I} \\
&= \frac{\iint X_\mathrm{I} p[Y(n, \mathrm{i}\omega_k)|X_\mathrm{R}, X_\mathrm{I}]p(X_\mathrm{R}, X_\mathrm{I})\,\mathrm{d}X_\mathrm{R}\,\mathrm{d}X_\mathrm{I}}{\iint p[Y(n, \mathrm{i}\omega_k)|X_\mathrm{R}, X_\mathrm{I}]p(X_\mathrm{R}, X_\mathrm{I})\,\mathrm{d}X_\mathrm{R}\,\mathrm{d}X_\mathrm{I}} .
\end{aligned} \tag{43.108}$$

The noise spectrum $V(n, \mathrm{i}\omega_k)$ is often assumed to be a complex Gaussian random variable. Now suppose that the real and imaginary parts of the speech spectrum can also be modeled as a Gaussian distribution. Let

$$\sigma_x^2 \triangleq E\left[|X(n, \mathrm{i}\omega_k)|^2\right]$$

and

$$\sigma_v^2 \triangleq E\left[|V(n, \mathrm{i}\omega_k)|^2\right]$$

denote, respectively, the variances of the clean speech and noise spectra. The pdf of the speech spectrum is given by

$$p(X_\mathcal{C}) = \frac{1}{\sqrt{\pi\sigma_x^2}} \exp\left(-\frac{X_\mathcal{C}^2}{\sigma_x^2}\right) , \quad \mathcal{C} \in \{\mathrm{R}, \mathrm{I}\} . \tag{43.109}$$

In such a condition, the conditional pdf $p[Y(n, \mathrm{i}\omega_k)|X_\mathrm{R}, X_\mathrm{I}]$ and the joint pdf $p(X_\mathrm{R}, X_\mathrm{I})$ can be expressed as

$$\begin{aligned}
p[Y(n, \mathrm{i}\omega_k)|X_\mathrm{R}, X_\mathrm{I}] &= p[V(n, \mathrm{i}\omega_k)] \\
&= \frac{1}{\pi\sigma_v^2} \exp\left[-\frac{|V(n, \mathrm{i}\omega_k)|^2}{\sigma_v^2}\right]
\end{aligned} \tag{43.110}$$

and

$$\begin{aligned}
p(X_\mathrm{R}, X_\mathrm{I}) &= p(X_\mathrm{R})p(X_\mathrm{I}) \\
&= \frac{1}{\pi\sigma_x^2} \exp\left[-\frac{|X(n, \mathrm{i}\omega_k)|^2}{\sigma_x^2}\right] .
\end{aligned} \tag{43.111}$$

Substituting (43.110) and (43.111) into (43.107) and (43.108), with some simple mathematical manipulation, we can deduce that

$$\widehat{X}_{\mathrm{MMSE}}(n, \mathrm{i}\omega_k) = \frac{\sigma_x^2}{\sigma_x^2 + \sigma_v^2} Y(n, \mathrm{i}\omega_k) . \tag{43.112}$$

If we introduce the concept of narrow-band a priori SNR, defined as

$$\zeta_k \triangleq \frac{\sigma_x^2}{\sigma_v^2} , \tag{43.113}$$

we can write (43.112) in the following form:

$$\begin{aligned}
\widehat{X}_{\mathrm{MMSE}}(n, \mathrm{i}\omega_k) &= H_{\mathrm{MMSE}}(\omega_k)Y(n, \mathrm{i}\omega_k) \\
&= \frac{\zeta_k}{1 + \zeta_k} Y(n, \mathrm{i}\omega_k) .
\end{aligned} \tag{43.114}$$

The form of (43.114) is identical to that of the estimate [given in (43.93)] obtained with the Wiener filter. This is expected since for Gaussian variables, the MMSE waveform estimator is always a linear function of the noisy observation [43.47], and so the two optimal MMSE solutions coincide.

In the above derivation, we assumed that the real and imaginary parts of the STFT coefficients of both the clean speech and noise signals can be modeled by a Gaussian distribution. This assumption generally holds for noise and approximately holds for speech when the analysis frame size is large and the span of the correlation of the speech signal is much shorter than the frame size. However, in the noise-reduction application, the frame size is typically selected between 10 and 40 ms.

Many studies have investigated the use of non-Gaussian distributions and found that the real and imaginary parts of the STFT coefficients of speech are better modeled with either the Gamma or Laplacian probability distributions when the frame size is short. In the Gamma case, the pdf of $X_{\mathcal{C}}$ [$\mathcal{C} \in \{\mathrm{R}, \mathrm{I}\}$] is given by

$$p(X_{\mathcal{C}}) = \frac{1}{2\sqrt{\pi}} \left(\frac{3}{2\sigma_x^2}\right)^{1/4} \frac{1}{\sqrt{|X_{\mathcal{C}}|}} \exp\left(-\sqrt{\frac{3}{2\sigma_x^2}}|X_{\mathcal{C}}|\right) . \tag{43.115}$$

In the Laplacian case, we have

$$p(X_{\mathcal{C}}) = \frac{1}{\sqrt{\sigma_x^2}} \exp\left(-\frac{2|X_{\mathcal{C}}|}{\sigma_x^2}\right) . \tag{43.116}$$

If we define the instantaneous narrow-band a posteriori SNR as

$$\gamma_{\mathcal{C},k} \triangleq \frac{Y_{\mathcal{C}}^2}{\sigma_v^2} , \tag{43.117}$$

where $Y_{\mathcal{C}}$ is defined in a similar way to $X_{\mathcal{C}}$, the MMSE estimator for $X_{\mathcal{C}}$ in both Gamma- and Laplacian-distribution conditions can be written in the form

$$\widehat{X}_{\mathcal{C},\mathrm{MMSE}} = H(\zeta_k, \gamma_{\mathcal{C},k}) Y_{\mathcal{C}} , \tag{43.118}$$

where $H(\zeta_k, \gamma_{\mathcal{C},k})$ is a gain function. For the Gamma distribution, the gain function is [43.48]

$$\begin{aligned} H(\zeta_k, \gamma_{\mathcal{C},k}) &= \left[\exp\left(\frac{\epsilon_{\mathcal{C}-}^2}{4}\right) D_{-1.5}(\epsilon_{\mathcal{C}-}) - \exp\left(\frac{\epsilon_{\mathcal{C}+}^2}{4}\right) D_{-1.5}(\epsilon_{\mathcal{C}+})\right] \\ &\Big/ \left\{(\epsilon_{\mathcal{C}+} - \epsilon_{\mathcal{C}-})\left[\exp\left(\frac{\epsilon_{\mathcal{C}-}^2}{4}\right) D_{-1.5}(\epsilon_{\mathcal{C}-}) + \exp\left(\frac{\epsilon_{\mathcal{C}+}^2}{4}\right) D_{-1.5}(\epsilon_{\mathcal{C}+})\right]\right\} , \end{aligned} \tag{43.119}$$

where

$$\epsilon_{\mathcal{C}\pm} \triangleq \frac{\sqrt{3}}{2\sqrt{\zeta_k}} \pm \sqrt{2\gamma_{\mathcal{C},k}} , \tag{43.120}$$

and $D_r(z)$ is the parabolic cylinder function [43.49]. For the Laplacian speech model, the gain function is written as [43.50, 51]

$$H(\zeta_k, \gamma_{\mathcal{C},k}) = \frac{2\left[\kappa_{\mathcal{C}+}\mathrm{erfcx}(\kappa_{\mathcal{C}+}) - \kappa_{\mathcal{C}-}\mathrm{erfcx}(\kappa_{\mathcal{C}-})\right]}{(\kappa_{\mathcal{C}+} - \kappa_{\mathcal{C}-})\left[\mathrm{erfcx}(\kappa_{\mathcal{C}+}) + \mathrm{erfcx}(\kappa_{\mathcal{C}-})\right]} , \tag{43.121}$$

where

$$\kappa_{\mathcal{C}\pm} \triangleq \frac{1}{\sqrt{\zeta_k}} \pm \sqrt{\gamma_{\mathcal{C},k}} , \tag{43.122}$$

and $\mathrm{erfcx}(z)$ is the scaled complementary error function, defined as

$$\mathrm{erfcx}(z) \triangleq \frac{2}{\sqrt{\pi}} \exp(z^2) \int_z^\infty \exp(-t^2)\,\mathrm{d}t . \tag{43.123}$$

Combining the real and imaginary spectral estimators, we can easily construct the MMSE estimator of the clean speech for both Gamma and Laplacian distributions.

43.5.2 MMSE Spectral Amplitude and Phase Estimators

The previous section discussed an MMSE estimator that estimates the real and imaginary spectral components and then combines the results together to form an estimate of the clean speech spectrum. Now, in this section, we take a slightly different perspective and address MMSE estimators of amplitude and phase spectra.

Consider the decomposition of the complex spectrum into amplitude and phase components, i. e.,

$$Y(n, \mathrm{i}\omega_k) = Y_k \exp\left(\mathrm{i}\theta_{Y_k}\right) , \tag{43.124a}$$

$$X(n, \mathrm{i}\omega_k) = X_k \exp\left(\mathrm{i}\theta_{X_k}\right) , \tag{43.124b}$$

where Y_k and X_k denote, respectively, the amplitudes of the noisy and clean speech spectra, and θ_{Y_k} and θ_{X_k} are their corresponding phase components. The problem of noise reduction then becomes one of designing two signal estimators that make decisions separately on the magnitude and phase spectra from the observed signal.

The MMSE estimator of the spectral amplitude X_k is given by the conditional mean

$$\begin{aligned} \widehat{X}_{k,\mathrm{MMSE}} &= \int_0^\infty X_k\, p[X_k | Y(n, \mathrm{i}\omega_k)]\,\mathrm{d}X_k \\ &= \frac{\int_0^\infty \int_0^\pi X_k\, p\left[Y(n,\mathrm{i}\omega_k)|X_k,\theta_{X_k}\right] p\left(X_k,\theta_{X_k}\right)\mathrm{d}X_k\,\mathrm{d}\theta_{X_k}}{\int_0^\infty \int_0^\pi p\left[Y(n,\mathrm{i}\omega_k)|X_k,\theta_{X_k}\right] p\left(X_k,\theta_{X_k}\right)\mathrm{d}X_k\,\mathrm{d}\theta_{X_k}} . \end{aligned} \tag{43.125}$$

If both noise and speech spectra are modeled as a Gaussian distribution, the conditional pdf $p[Y(n, \mathrm{i}\omega_k)|X_k, \theta_{X_k}]$ can easily be deduced as

$$p\left[Y(n, \mathrm{i}\omega_k)|X_k, \theta_{X_k}\right] = \frac{1}{\pi\sigma_v^2} \exp\left[-\frac{|V(n, \mathrm{i}\omega_k)|^2}{\sigma_v^2}\right] . \tag{43.126}$$

It is known that, for complex Gaussian random variables with zero mean, their amplitude and phase components are statistically independent [43.47]. Therefore, we have

$$p(X_k, \theta_{X_k}) = p(X_k)p(\theta_{X_k}) , \tag{43.127}$$

where $p(X_k)$ is a Rayleigh density,

$$p(X_k) = \frac{2X_k}{\sigma_x^2} \exp\left(-\frac{X_k^2}{\sigma_x^2}\right) , \tag{43.128}$$

and $p(\theta_{X_k})$ is a uniform density with

$$p(\theta_{X_k}) = \frac{1}{2\pi} . \tag{43.129}$$

Substituting (43.126) and (43.127) into (43.125) gives

$$\widehat{X}_{k,\mathrm{MMSE}} = \frac{\int_0^\infty X_k^2 \exp\left[-\frac{1}{\sigma_v^2}\left(Y_k^2 + \frac{\sigma_x^2+\sigma_v^2}{\sigma_v^2}\right)\right] I_0\left(\frac{2X_kY_k}{\sigma_v^2}\right) \mathrm{d}X_k}{\int_0^\infty X_k \exp\left[-\frac{1}{\sigma_v^2}\left(Y_k^2 + \frac{\sigma_x^2+\sigma_v^2}{\sigma_v^2}\right)\right] I_0\left(\frac{2X_kY_k}{\sigma_v^2}\right) \mathrm{d}X_k} . \tag{43.130}$$

One can check that the numerator and denominator of (43.130) are in the form of the second- and first-order moments, respectively, of the Rician density function [43.52, 53].

With some mathematical manipulation, the estimator (43.130) can be expressed in the following form [43.27],

$$\widehat{X}_{k,\mathrm{MMSE}} = \Gamma\left(\frac{3}{2}\right)\frac{\sqrt{\vartheta_k}}{\gamma_k}\exp\left(-\frac{\vartheta_k}{2}\right) \times\left[(1+\vartheta_k)I_0\left(\frac{\vartheta_k}{2}\right)+\vartheta_k I_1\left(\frac{\vartheta_k}{2}\right)\right]Y_k , \tag{43.131}$$

where $\Gamma(\cdot)$ denotes the gamma function, with $\Gamma(3/2) = \sqrt{\pi}/2$, $I_0(\cdot)$ and $I_1(\cdot)$ are, respectively, the modified Bessel functions of zero and first order, and

$$\vartheta_k \triangleq \frac{\zeta_k}{1+\zeta_k}\gamma_k \tag{43.132}$$

with

$$\zeta_k \triangleq \frac{\sigma_x^2}{\sigma_v^2} = \frac{P_x(\omega_k)}{P_v(\omega_k)} \tag{43.133}$$

and

$$\gamma_k \triangleq \frac{Y_k^2}{\sigma_v^2} , \tag{43.134}$$

denoting, respectively, the subband a priori and the instantaneous subband a posteriori SNR [43.27].

Now let us estimate the phase component of the clean speech spectrum from the noisy speech spectrum. Ephraim and Malah formulated an MMSE complex exponential estimator [43.27], i. e.,

$$\left[\exp\left(\mathrm{i}\hat{\theta}_{X_k}\right)\right]_{\mathrm{MMSE}} = \arg\min_{\exp\left(\mathrm{i}\hat{\theta}_{X_k}\right)} E\left[\left|\exp\left(\mathrm{i}\theta_{X_k}\right)-\exp\left(\mathrm{i}\hat{\theta}_{X_k}\right)\right|^2\right] . \tag{43.135}$$

It turns out that the solution of this MMSE estimator is

$$\left[\exp\left(\mathrm{i}\hat{\theta}_{X_k}\right)\right]_{\mathrm{MMSE}} = \exp\left(\mathrm{i}\theta_{Y_k}\right) . \tag{43.136}$$

That is, the exponential of the noisy phase θ_{Y_k} is the MMSE exponential estimator of the signal's phase.

Also derived in [43.27] was an optimal estimator of the principle value of the signal's phase, i. e.,

$$\hat{\theta}_{X_k,\mathrm{o}} = \arg\min_{\hat{\theta}_{X_k}} E\left[1-\cos\left(\theta_{X_k}-\hat{\theta}_{X_k}\right)\right] . \tag{43.137}$$

The solution of this estimator is

$$\hat{\theta}_{X_k,\mathrm{o}} = \theta_{Y_k} , \tag{43.138}$$

which indicates that the noisy phase is an optimal estimate of the principle value of the signal's phase.

Combining the amplitude and phase estimators, we can construct an estimate of the clean speech spectrum as

$$\widehat{X}_{\mathrm{MMSE}}(n, \mathrm{i}\omega_k) = \widehat{X}_{k,\mathrm{MMSE}}\exp\left(\mathrm{i}\theta_{Y_k}\right) = H_{\mathrm{MMSE}}(\zeta_k, \gamma_k)Y(n, \mathrm{i}\omega_k) , \tag{43.139}$$

where the gain function $H_{\mathrm{MMSE}}(\zeta_k, \gamma_k)$ is

$$H_{\mathrm{MMSE}}(\zeta_k, \gamma_k) = \Gamma\left(\frac{3}{2}\right)\frac{\sqrt{\vartheta_k}}{\gamma_k}\exp\left(-\frac{\vartheta_k}{2}\right) \times\left[(1+\vartheta_k)I_0\left(\frac{\vartheta_k}{2}\right)+\vartheta_k I_1\left(\frac{\vartheta_k}{2}\right)\right] . \tag{43.140}$$

In high-SNR conditions (i. e., $\zeta_k \gg 1$), (43.140) can be simplified as

$$H_{\mathrm{MMSE}}(\zeta_k, \gamma_k) \approx \frac{\zeta_k}{1+\zeta_k} . \tag{43.141}$$

In such a situation, the MMSE estimate of the clean speech spectrum can be written as

$$\hat{X}_{\mathrm{MMSE}}(n, \mathrm{i}\omega_k) = \frac{\zeta_k}{1+\zeta_k}Y(n, \mathrm{i}\omega_k) = \frac{P_x(\omega_k)}{P_y(\omega_k)}Y(n, \mathrm{i}\omega_k) . \tag{43.142}$$

This result is the same as the frequency-domain Wiener filter. Therefore, the MMSE amplitude estimator converges to the Wiener filter in high-SNR conditions.

43.5.3 Maximum A Posteriori (MAP) Spectral Estimator

We have discussed two MMSE spectral estimators: one optimizes the real and imaginary parts of the clean speech spectrum separately, and the other deals with spectral amplitude and phase separately. We now consider the joint estimation of the spectral amplitude and phase using the MAP criterion. The MAP estimator can be written as

$$\widehat{X}_{k,\mathrm{MAP}} = \arg\max_{X_k} J_{\mathrm{MAP}}\left(X_k, \theta_{X_k}\right) , \tag{43.143}$$

and

$$\hat{\theta}_{X_k,\mathrm{MAP}} = \arg\max_{\theta_{X_k}} J_{\mathrm{MAP}}\left(X_k, \theta_{X_k}\right) , \tag{43.144}$$

where the MAP cost function is expressed as

$$J_{\mathrm{MAP}}(X_k, \theta_{X_k}) = \ln\left\{p\left[X_k, \theta_{X_k}\middle|Y(n, \mathrm{i}\omega)\right]\right\} . \tag{43.145}$$

For the optimization problems given in (43.143) and (43.144), the MAP cost function above can be equivalently written as

$$\begin{aligned} &J_{\mathrm{MAP}}\left(X_k, \theta_{X_k}\right) \\ &= \ln\left\{p\left[Y(n, \mathrm{i}\omega)\middle|X_k, \theta_{X_k}\right]\right] p\left(X_k, \theta_{X_k}\right)\right\} . \end{aligned} \tag{43.146}$$

If we assume that both speech and noise spectra can be modeled by Gaussian distribution, according to (43.126) and (43.127), we can readily deduce that

$$\begin{aligned} &p\left[Y(n, \mathrm{i}\omega)\middle|X_k, \theta_{X_k}\right]\right] p(X_k, \theta_{X_k}) \\ &= \frac{X_k}{\pi^2\sigma_x^2\sigma_v^2} \\ &\quad \times \exp\left[-\frac{X_k^2}{\sigma_x^2} - \frac{\left|Y(n, \mathrm{i}\omega_k) - X_k \exp\left(\mathrm{i}\theta_{X_k}\right)\right|^2}{\sigma_v^2}\right] . \end{aligned} \tag{43.147}$$

Substituting (43.147) into (43.146), differentiating the MAP cost function with respect to X_k, and equating the result to zero, we obtain [43.30]

$$\widehat{X}_{k,\mathrm{MAP}} = \frac{\zeta_k + \sqrt{\zeta_k^2 + 2(1+\zeta_k)\zeta_k/\gamma_k}}{2(1+\zeta_k)} Y_k , \tag{43.148}$$

where ζ_k and γ_k denote, respectively, the subband a priori and instantaneous subband a posteriori SNR as defined in Sect. 43.5.1.

Similarly, if we substitute (43.147) into (43.146), differentiate the MAP cost function with respect to θ_X, and equate the result to zero, we obtain

$$\hat{\theta}_{X_k,\mathrm{MAP}} = \theta_{Y_k} . \tag{43.149}$$

So, this MAP estimate of the speech phase, which is identical to the MMSE phase estimator, is the phase of the noisy speech.

Combining (43.148) and (43.149), we achieve the MAP estimate of the clean speech spectrum

$$\widehat{X}_{\mathrm{MAP}}(n, \mathrm{i}\omega_k) = H_{\mathrm{MAP}}(\omega_k) Y(n, \mathrm{i}\omega_k) , \tag{43.150}$$

where

$$H_{\mathrm{MAP}}(\omega_k) = \frac{\zeta_k + \sqrt{\zeta_k^2 + 2(1+\zeta_k)\zeta_k/\gamma_k}}{2(1+\zeta_k)} . \tag{43.151}$$

When $\zeta_k \gg 1$, we can approximately have

$$H_{\mathrm{MAP}}(\omega_k) \approx \frac{\zeta_k}{1+\zeta_k} . \tag{43.152}$$

Therefore, in high-SNR conditions, this MAP estimator will be the same as the MMSE estimator and the Wiener filter.

43.5.4 Maximum-Likelihood Spectral Amplitude Estimator

We have seen from previous analysis that the noise-reduction problem is essentially a matter of recovering the spectral amplitude since the optimal estimate of the signal's phase turned out to be the noisy phase. One may still want to ask: can noise reduction performance be augmented by recovering the signal's phase? To answer this question, many perception experiments have been conducted and the results have shown that the human perception system is relatively insensitive to phase corruption. At least, within the context of noise reduction, it has been shown that speech distortion resulting from phase corruption is generally imperceptible when subband SNR at any ω_k is greater than about 6 dB [43.15]. In addition, *Wang* and *Lim* [43.54] compared the degree of distortion to reconstructed speech using noisy estimates of spectral magnitude and phase. Using listening tests, their results showed that the subjective quality of speech increases dramatically with an increase in the accuracy of the magnitude estimate, whereas speech quality increases only marginally with increases in the accuracy of the phase estimate.

From both the signal-processing and perception points of view, we see that the use of noisy phase as

the signal's phase is good enough for speech enhancement in most applications. As a result, the single-channel noise-reduction problem is generally formulated to acquire an estimate of the amplitude of the clean speech spectrum. Now let us consider how to estimate the amplitude of the clean speech spectrum. One straightforward way of doing this is through spectral substraction [43.12] or parametric spectral subtraction [43.17]. These techniques will be extensively discussed in the following chapter. Here we consider the maximum likelihood (ML) method, which is one of the most popularly used statistical estimators due to its asymptotic optimal property, i. e., the estimation variance can achieve the Cramèr–Rao lower bound (CRLB) when the number of observation samples approaches infinity.

Let us rewrite the signal model given in (43.3) in the form

$$\begin{aligned} Y(n,\mathrm{i}\omega_k) &= X(n,\mathrm{i}\omega_k)+V(n,\mathrm{i}\omega_k) \\ &= X_k\exp\left(\mathrm{i}\theta_{X_k}\right)+V(n,\mathrm{i}\omega_k)\,. \end{aligned} \tag{43.153}$$

Then the ML estimator for X_k can be written as

$$\widehat{X}_{k,\mathrm{ML}} = \arg\max_{X_k} \mathcal{L}\,, \tag{43.154}$$

where

$$\begin{aligned} \mathcal{L} &= \ln\left\{p\left[Y(n,\mathrm{i}\omega_k)|X_k\right]\right\} \\ &= \ln\left\{\int_{-\pi}^{\pi} p\left[Y(n,\mathrm{i}\omega_k)|X_k,\theta_{X_k}\right]p\left(\theta_{X_k}\right)\,\mathrm{d}\theta_{X_k}\right\} \end{aligned} \tag{43.155}$$

is the log-likelihood cost function. If the noise spectrum $V(n,\mathrm{i}\omega_k)$ is assumed to be a Gaussian process, the conditional pdf $p[Y(n,\mathrm{i}\omega_k)|X_k,\theta_{X_k}]$ is given by (43.126). Substituting (43.126) into (43.155) and assuming that θ_{X_k} is uniformly distributed between $-\pi$ and π, we can deduce that

$$\begin{aligned} &p\left[Y(n,\mathrm{i}\omega_k)|X_k\right] \\ &= \frac{1}{\pi\sigma_v^2}\exp\left[-\frac{Y_k^2+X_k^2}{\sigma_v^2}\right]I_0\left[\frac{2X_kY_k}{\sigma_v^2}\right]\,. \end{aligned} \tag{43.156}$$

Substituting (43.156) into (43.155), differentiating the log-likelihood cost function with respect to X_k, and equating the result to zero, we can obtain the ML estimate of the spectral amplitude. Unfortunately, the above ML estimator is difficult to implement since the Bessel function is not reducible. However, the modified Bessel function of order zero, when its argument is large, can be approximated as

$$I_0(|\alpha|) = \frac{1}{\sqrt{2\pi|\alpha|}}\exp(|\alpha|)\,,\quad \text{for } |\alpha|\gg 1\,. \tag{43.157}$$

So, when $2X_kY_k/\sigma_v^2 \gg 1$ (this is equivalent to saying that the a priori SNR is high), the conditional pdf $p\left[Y(n,\mathrm{i}\omega_k)|X_k,\theta_X\right]$ can be approximated as

$$\begin{aligned} &p[Y(n,\mathrm{i}\omega_k)|X_k] \\ &\approx \frac{1}{2\pi\sigma_v\sqrt{\pi X_kY_k}}\exp\left(-\frac{Y_k^2-2X_kY_k+X_k^2}{\sigma_v^2}\right)\,. \end{aligned} \tag{43.158}$$

Substituting this approximation into (43.155), we can readily derive an approximate ML estimator for the spectral amplitude, i. e.,

$$\widehat{X}_{k,\mathrm{MLA}} = \frac{Y_k+\sqrt{Y_k^2-\sigma_v^2}}{2}\,. \tag{43.159}$$

In this case, the estimate of the clean speech spectrum can be expressed as

$$\begin{aligned} \hat{X}_{\mathrm{MLA}}(n,\mathrm{i}\omega_k) &= \widehat{X}_{k,\mathrm{MLA}}\exp\left(\theta_{Y_k}\right) \\ &= H_{\mathrm{MLA}}(\omega_k)Y(n,\mathrm{i}\omega_k)\,, \end{aligned} \tag{43.160}$$

where

$$H_{\mathrm{MLA}}(\omega_k) = \frac{1+\sqrt{(Y_k^2-\sigma_v^2)/Y_k^2}}{2}$$

is a gain filter.

43.5.5 Maximum-Likelihood Spectral Power Estimator

The ML method can also be applied to estimate the spectral power. For the signal model given in (43.3), if we assume both the clean and noise signals are Gaussian processes, then $Y(n,\mathrm{i}\omega_k)$ is also a Gaussian random variable whose pdf is given by

$$p[Y(n,\mathrm{i}\omega_k)] = \frac{1}{\pi\left(\sigma_x^2+\sigma_v^2\right)}\exp\left(-\frac{Y_k^2}{\sigma_x^2+\sigma_v^2}\right)\,. \tag{43.161}$$

The log-likelihood function of $Y(n,\mathrm{i}\omega_k)$, given the speech power σ_x^2 and noise power σ_v^2, can be written as

$$\begin{aligned} \mathcal{L} &= \ln\left\{p\left[Y(n,\mathrm{i}\omega_k)|\sigma_x^2,\sigma_v^2\right]\right\} \\ &= -\ln(\pi)-\ln\left(\sigma_x^2+\sigma_v^2\right)-\frac{Y_k^2}{\sigma_x^2+\sigma_v^2}\,. \end{aligned} \tag{43.162}$$

Differentiating the log-likelihood function $\mathcal{L}$ with respect to σ_x^2 and equating the result to zero, we can readily derive the ML spectral power estimator,

$$\hat{\sigma}^2_{X,\mathrm{ML}} = Y_k^2 - \sigma_v^2 \,. \tag{43.163}$$

An estimate of the clean speech spectrum can then be constructed as

$$\begin{aligned}\hat{X}_{\mathrm{MLP}}(n, \mathrm{i}\omega_k) &= \sqrt{\hat{\sigma}^2_{X,\mathrm{ML}}}\exp\left(\mathrm{i}\theta_{Y_k}\right) \\ &= H_{\mathrm{MLP}}(\omega_k)Y(n, \mathrm{i}\omega_k)\,,\end{aligned} \tag{43.164}$$

where

$$H_{\mathrm{MLP}}(\omega_k) = \sqrt{\frac{Y_k^2 - \sigma_v^2}{Y_k^2}} \tag{43.165}$$

is a gain filter.

43.5.6 MAP Spectral Amplitude Estimator

In the previous subsections, we developed two ML estimators that assume the clean speech spectrum is fixed but unknown. We now depart from this philosophy to investigate a particular maximum a posteriori (MAP) estimator, namely,

$$\widehat{X}_{k,\,\mathrm{MAP}} = \arg\max_{X_k} J_{\mathrm{MAP}}(X_k)\,, \tag{43.166}$$

where the MAP cost function is written as

$$\begin{aligned}&J_{\mathrm{MAP}}(X_k) \\ &= \ln\{p[X_k|Y(n, \mathrm{i}\omega)]\} \\ &= \ln\left\{\int_{-\pi}^{\pi} p\left[Y(n, \mathrm{i}\omega)|X_k, \theta_{X_k}\right]p\left(X_k, \theta_{X_k}\right)p\left(\theta_{X_k}\right)\mathrm{d}\theta_{X_k}\right\}.\end{aligned} \tag{43.167}$$

Following (43.126) and (43.127), if we define

$$\rho_k = \frac{\sigma_x^2\sigma_v^2}{2\left(\sigma_x^2 + \sigma_v^2\right)} \tag{43.168}$$

and

$$\varrho_k = \sqrt{\rho_k \vartheta_k}\,, \tag{43.169}$$

where ϑ_k is defined in (43.132), we find that

$$\begin{aligned}&p[X_k|Y(n, \mathrm{i}\omega)] \\ &= \frac{X_k}{\rho_k}\exp\left(-\frac{X_k^2 + \varrho_k^2}{2\rho_k}\right) I_0\left(\frac{X_k\varrho_k}{\rho_k}\right)\,.\end{aligned} \tag{43.170}$$

Using the approximation given in (43.157), we can further write (43.170) in the form

$$p[X_k|Y(n, \mathrm{i}\omega)] \approx \frac{\sqrt{X_k}}{\sqrt{2\pi\rho_k\varrho_k}}\exp\left[-\frac{(X_k - \varrho_k)^2}{2\rho_k}\right]\,. \tag{43.171}$$

Substituting (43.171) into (43.167), differentiating the cost function with respect to X_k, and equating the result to zero, we find

$$\widehat{X}_{k,\mathrm{MAP}} = \frac{\zeta_k + \sqrt{\zeta_k^2 + (1+\zeta_k)\zeta_k/\gamma_k}}{2(1+\zeta_k)}Y_k\,, \tag{43.172}$$

where ζ_k is the frequency-domain subband a priori SNR (43.133) and γ_k is the instantaneous subband a posteriori SNR (43.133). Together with the noisy phase, we can construct an estimate of the clean speech spectrum via the approximate MAP method as

$$\widehat{X}_{\mathrm{MAPA}}(n, \mathrm{i}\omega_k) = H_{\mathrm{MAPA}}(\omega_k)Y(n, \mathrm{i}\omega_k)\,, \tag{43.173}$$

where

$$H_{\mathrm{MAPA}}(\omega_k) = \frac{\zeta_k + \sqrt{\zeta_k^2 + (1+\zeta_k)\zeta_k/\gamma_k}}{2(1+\zeta_k)}\,. \tag{43.174}$$

43.6 Speech-Model-Based Noise Reduction

From Von Kempelen's and Wheatstone's speaking machines, to *Stewart*'s vocoder, to *Dunn*'s electrical vocal tract, and then to *Atal* and *Itakura*'s linear prediction (LP) technique [43.55–60], it took scientists nearly two centuries to develop mathematical models that can represent human speech production and characterize the changing characteristics of speech. The discovery of speech models, particularly discrete ones such as the harmonic model, LP model, and hidden Markov model (HMM), has opened a new era for speech processing. For example, with the use of discrete speech models, noise reduction can be converted into a parameter-estimation problem in a much lower-dimensional space. In this way, we can fully take advantage of our knowledge of speech production to enhance a speech signal corrupted by additive noise.

43.6.1 Harmonic-Model-Based Noise Reduction

The harmonic (or sinusoidal) model has been applied with certain success to many speech processing problems [43.14, 61–66]. In this model, the speech signal is represented as a sum of harmonically related sine waves with frequencies given by pitch harmonics. Specifically, at time n, the speech signal $x(n)$ is represented as

$$x(n) = \sum_{i=1}^{Q} A_i(n) \cos(2\pi f_i n + \phi_i) \,, \tag{43.175}$$

where $A_i(n)$, f_i, and ϕ_i are, respectively, the time-varying amplitude, frequency, and initial phase of the i-th tone, and Q denotes the total number of harmonics. For voiced speech, the f_i are multiples of the fundamental frequency f_0, i. e., $f_i = i f_0$. Hence the model (43.175) reduces to

$$x(n) = \sum_{i=1}^{Q} A_i(n) \cos(2\pi i f_0 n + \phi_i) \,. \tag{43.176}$$

With the above speech source model, the microphone observation signal $y(n)$ can be decomposed into two parts: harmonics of speech, and noise. Since speech has energy only at harmonics, while noise, in general, has energy in the frequency regions between harmonics, comb filtering can be applied to reduce noise while preserving the speech periodicity, provided that the information of the fundamental frequency is known. In other words, an estimate of the clean speech can be obtained as [43.67]

$$\hat{x}(n) = y(n) * h_{\mathrm{COMB}}(n) \,, \tag{43.177}$$

where

$$h_{\mathrm{COMB}}(n) = \sum_{i=-K}^{K} h_i \delta(n - N_i) \tag{43.178}$$

is the impulse response of the comb filter, $\delta(n)$ denotes the unit impulse function, h_i are the filter coefficients, which satisfy $\sum_{i=-K}^{K} h_i = 1$,

$$N_i = \begin{cases} -\sum_{j=0}^{i-1} T_j \,, & i > 0 \\ 0 \,, & i = 0 \\ -\sum_{j=i}^{-1} T_j \,, & i < 0 \end{cases} \tag{43.179}$$

and T_j is the instantaneous pitch period, which contains the point of the speech waveform that is multiplied by the filter coefficient h_i. It is seen that the length of the comb filter is $2K+1$ pitch periods.

With accurate and reliable algorithms to update both the filter coefficients h_i and the time-varying pitch T_j, the comb filtering operation given in (43.177) can significantly attenuate the background noise presented in the noisy speech signal. However, one major drawback with this approach is that comb filtering should be restricted only to vowels or vowel-like sounds. If applied to the unvoiced speech, it usually introduces audible speech distortion, which is undesirable. Many efforts have been devoted to improving this method, such as dealing with voiced and unvoiced speech separately, or using a more accurate harmonic-plus-noise model (HNM). But, in general, this method suffers higher speech distortion than filtering, spectral restoration, and other model-based noise-reduction techniques.

43.6.2 Linear-Prediction-Based Noise Reduction

The basic idea underlying the linear prediction (LP) model [also called the autoregressive (AR) model] is that, during a short-time stationary frame of speech, the production system can be described by an all-pole system driven by an excitation sequence [43.41, 59, 60, 68]. The excitation can be either a pulse train (for voiced speech) or random noise (for unvoiced sound). For ease of analysis, let us first assume that the excitation is random noise. In this situation, the speech signal $x(n)$, at time n can be written as

$$\begin{aligned} x(n) &= \sum_{p=1}^{P} a_{x,p} x(n-p) + g_x u(n) \\ &= \boldsymbol{a}_x^{\mathrm{T}} \boldsymbol{x}(n-1) + g_x u(n) \,, \end{aligned} \tag{43.180}$$

where $a_{x,p}$ $(p = 1, 2, \cdots, P)$ are the LP coefficients, which are assumed constant over the analysis speech frame, P is the order of the linear prediction, g_x represents the gain of the excitation, $u(n)$ is a Gaussian random process with zero mean and unit variance, and

$$\begin{aligned} \boldsymbol{a}_x &= [\, a_{x,1} \;\; a_{x,2} \;\; \cdots \;\; a_{x,P} \,]^{\mathrm{T}} \,, \\ \boldsymbol{x}(n-1) &= [\, x(n-1) \;\; x(n-2) \;\; \cdots \;\; x(n-P) \,]^{\mathrm{T}} \,. \end{aligned} \tag{43.181}$$

It is seen from (43.180) that the clean speech $x(n)$ depends on a total of $2P+1$ parameters, namely the P coefficients in the prediction vector $\boldsymbol{a}_x$, the single gain factor g_x, and the P speech samples in the initial $\boldsymbol{x}$ vector; we denote this initial vector by $\boldsymbol{x}(0)$. With this LP

speech model, the microphone signal observed in the presence of noise can be written as

$$y(n) = x(n) + v(n) = \boldsymbol{a}_x^{\mathrm{T}} \boldsymbol{x}(n-1) + g_x u(n) + v(n) \,. \tag{43.182}$$

It becomes immediately clear that the noise-reduction problem can be formulated as one of estimating the parameters $\boldsymbol{a}_x$, g_x, and $\boldsymbol{x}(0)$ from the noisy observation $y(n)$. Relatively, the initial vector plays a less important role than the LP coefficients and gain factor, so we can simply assume that this vector consists of all zeros or the first P samples of the noisy speech signal. Therefore, the noise-reduction problem is indeed a matter of estimating the LP coefficients and gain factor, which together are called the LP model parameters.

Following the same line of ideas discussed in Sect. 43.5, we can formulate different estimators such as MMSE, ML, MAP, etc., to estimate the LP model parameters of the clean speech from the noisy observations. One widely investigated method is the sequential MAP estimator originally developed by *Lim* and *Oppenheim* [43.13, 69]. This method optimizes the LP model parameters and the clean speech estimate in an iterative way until some criterion is satisfied. Suppose that at time n we have a frame of observation signal $\boldsymbol{y}(n)$ and denote $\hat{\boldsymbol{x}}_i(n)$, $\hat{\boldsymbol{a}}_{x,i}$, and $\hat{g}_{x,i}$, respectively, as the estimate of the clean speech vector $\boldsymbol{x}(n)$, the LP vector $\boldsymbol{a}_x$, and the gain factor g_x from the ith iteration. The sequential MAP technique estimates the LP model parameters and the clean speech vector based on the following steps:

1. Obtain a MAP estimate of the LP model parameters based on the previous estimate of the model parameters and clean speech, i. e.,

$$\left(\hat{\boldsymbol{a}}_{x,i}, \hat{g}_{x,i}\right) = \arg\max_{\boldsymbol{a}_x, g_x} p\left[\boldsymbol{a}_x, g_x | \boldsymbol{y}(n), \hat{\boldsymbol{x}}_{i-1}, \boldsymbol{x}(0)\right] \,. \tag{43.183}$$

2. Obtain a MAP estimate of the clean speech with the newly estimated model parameters, i. e.,

$$\hat{\boldsymbol{x}}_i(n) = \arg\max_{\boldsymbol{x}(n)} p\left[\boldsymbol{x}(n) | \boldsymbol{y}(n), \hat{\boldsymbol{a}}_{x,i}, \hat{g}_{x,i}\right] \,. \tag{43.184}$$

With the assumption that all the unknown parameters are random with a Gaussian pdf, it can be shown that the first step is equivalent to estimating the LP model parameters from $\hat{\boldsymbol{x}}_{i-1}$ using the correlation method of linear prediction analysis, and the second step achieves the clean speech estimate through the optimal Wiener filter

$$H_{\mathrm{o},i}(\omega) = \frac{\hat{P}_x(\omega, i)}{\hat{P}_x(\omega, i) + \hat{P}_v(\omega, i)} \,, \tag{43.185}$$

where

$$\hat{P}_x(\omega, i) = \frac{g_x^2}{\left|1 - \hat{\boldsymbol{a}}_{x,i}^{\mathrm{T}} \boldsymbol{f}\right|^2} \,,$$
$$\boldsymbol{f} = [\exp(-\mathrm{i}\omega) \;\; \exp(-\mathrm{i}2\omega) \;\; \cdots \;\; \exp(-\mathrm{i}P\omega)]^{\mathrm{T}} \,, \tag{43.186}$$

and $\hat{P}_v(\omega, i) = \sigma_v^2$ if $v(n)$ is white Gaussian noise. We see that the second step is the same as the frequency-domain Wiener filter as discussed in Sect. 43.4.4. The only difference is that in Sect. 43.4.4, the power spectrum is estimated via periodogram, while here it is computed from the LP model. It has been shown that, if the above two steps are repeated until some specified error criterion is satisfied, this iterative technique can increase the joint likelihood of $\hat{\boldsymbol{a}}_{x,i}$ and $\hat{\boldsymbol{x}}_i(n)$ with each iteration, thereby provide an optimal estimator in the MMSE sense. The above method can also be easily extended to the colored-noise case [43.69].

Another intensively investigated estimation method is based on the Kalman filtering technique. Putting (43.180) into matrix/vector form, we get

$$\begin{aligned} \boldsymbol{x}(n) &= \boldsymbol{F}\boldsymbol{x}(n-1) + \boldsymbol{g}u(n) \,, \\ y(n) &= \boldsymbol{c}^{\mathrm{T}} \boldsymbol{x}(n) + v(n) \,, \end{aligned} \tag{43.187}$$

where

$$\boldsymbol{F} = \begin{pmatrix} a_{x,1} & a_{x,2} & \cdots & a_{x,P-1} & a_{x,P} \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \cdots & \cdots & \vdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix} \tag{43.188}$$

and

$$\boldsymbol{g} = \boldsymbol{c} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \end{bmatrix}^{\mathrm{T}} \,. \tag{43.189}$$

If we assume that all the parameters $a_{x,p}$ $(p = 1, 2, \cdots, P)$, g_x, and σ_v^2 are known, an optimal estimate of $\boldsymbol{x}(n)$ can be obtained using the Kalman filter [43.31]. Briefly, the solution of using Kalman filter can be sum-

marized as

$$\hat{\boldsymbol{x}}(n)=\boldsymbol{F}\hat{\boldsymbol{x}}(n-1)+\boldsymbol{k}(n)[y(n)-\boldsymbol{c}^{\mathrm{T}}\hat{\boldsymbol{x}}(n-1)]\,, \tag{43.190a}$$

$$\boldsymbol{k}(n)=\boldsymbol{M}(n|n-1)\boldsymbol{c}\left[\boldsymbol{c}^{\mathrm{T}}\boldsymbol{M}(n|n-1)\boldsymbol{c}+\sigma_v^2\right]^{-1}, \tag{43.190b}$$

$$\boldsymbol{M}(n|n-1)=\boldsymbol{F}\boldsymbol{M}(n-1)\boldsymbol{F}^{\mathrm{T}}+\sigma_u^2\boldsymbol{g}\boldsymbol{g}^{\mathrm{T}}\,, \tag{43.190c}$$

$$\boldsymbol{M}(n)=[\boldsymbol{I}-\boldsymbol{k}(n)\boldsymbol{c}^{\mathrm{T}}]\boldsymbol{M}(n|n-1)\,. \tag{43.190d}$$

where $\hat{\boldsymbol{x}}(n)$ is the estimate of $\boldsymbol{x}(n)$, $\boldsymbol{k}(n)$ is the Kalman gain vector, $\boldsymbol{M}(n|n-1)=E\{[\boldsymbol{x}(n)-\boldsymbol{F}\hat{\boldsymbol{x}}(n-1)][\boldsymbol{x}(n)-\boldsymbol{F}\hat{\boldsymbol{x}}(n-1)]^{\mathrm{T}}\}$ is the predicted state-error covariance matrix, and $\boldsymbol{M}(n|n)=E\{[\boldsymbol{x}(n)-\hat{\boldsymbol{x}}(n)][\boldsymbol{x}(n)-\hat{\boldsymbol{x}}(n)]^{\mathrm{T}}\}$ is the filtered state-error covariance matrix. After Kalman filtering, the estimate of the clean speech at time n is

$$\hat{x}(n)=\boldsymbol{c}^{\mathrm{T}}\hat{\boldsymbol{x}}(n)\,. \tag{43.191}$$

In this Kalman filtering technique, we make the following assumptions:

1. Within the analysis speech frame, both the LP coefficients and the excitation are constant.
2. The LP model parameters of the clean speech are known.
3. The excitation can be modeled as a Gaussian random process.
4. The observation noise $v(n)$ is a white Gaussian process.

This technique was first proposed by *Paliwal* and *Basu* in [43.31]. It can be shown that, when the aforementioned four assumptions hold, the method can significantly reduce the observation noise. However, the four assumptions can hardly be satisfied in practical environments, making the technique difficult to implement. Many efforts have been spent to relax some of these assumptions, thereby making the technique more practical. Broadly speaking, these efforts can be classified into four categories:

1. Assuming LP model parameters are time varying but can be modeled as stationary processes [43.70, 71]. The resulting methods are often called time-varying AR-based techniques, which are meant to remedy the first assumption.
2. The LP model parameters of the clean speech are normally not known in real applications. Several methods have been developed to estimate these parameters from the noisy speech. The most representative of these is the iterative or estimate-maximize algorithm [43.33, 72, 73], where the LP model and noise parameters are optimized iteratively, and the enhanced speech is obtained as a byproduct of the parameter estimation. Apparently, this group of efforts are to remedy the second assumption.
3. As we know, the excitation for unvoiced sound can be modeled as random noise, but it should be a pulse train for voiced speech. To model the excitation better, methods such as multiple variances [43.73], algebraic code excitation [43.74], and mixture Gaussian noise [43.73,75] have been developed. These are meant to remedy the third assumption.
4. In the case that $v(n)$ is not white but colored, we can model it with an LP model with a lower order than that of the speech, i. e.,

$$v(n)=\sum_{q=1}^{Q}a_{v,q}v(n-q)+g_v\eta(n)\,. \tag{43.192}$$

In matrix/vector form, (43.192) can be written as

$$\begin{aligned}\boldsymbol{v}(n)&=\boldsymbol{F}_v\boldsymbol{v}(n-1)+\boldsymbol{g}_v\eta(n)\,,\\ v(n)&=\boldsymbol{c}_v^{\mathrm{T}}\boldsymbol{v}(n)\,,\end{aligned} \tag{43.193}$$

where $\boldsymbol{F}_v$, $\boldsymbol{g}_v$, and $\boldsymbol{c}_v$ are defined similarly to their counterparts in (43.187). Now if (43.187) and (43.193) are combined, we get the augmented state equations:

$$\begin{aligned}\tilde{\boldsymbol{x}}(n)&=\tilde{\boldsymbol{F}}\tilde{\boldsymbol{x}}(n-1)+\tilde{\boldsymbol{G}}\tilde{\boldsymbol{u}}(n)\,,\\ y(n)&=\tilde{\boldsymbol{c}}^{\mathrm{T}}\tilde{\boldsymbol{x}}(n)\,,\end{aligned} \tag{43.194}$$

where

$$\tilde{\boldsymbol{x}}(n)=[\,\boldsymbol{x}^{\mathrm{T}}(n)\;\;\boldsymbol{v}^{\mathrm{T}}(n)\,]^{\mathrm{T}}\,.$$

$$\tilde{\boldsymbol{F}}=\begin{pmatrix}\boldsymbol{F}&\boldsymbol{0}\\ \boldsymbol{0}&\boldsymbol{F}_v\end{pmatrix}\,,$$

$$\tilde{\boldsymbol{G}}=\begin{pmatrix}\boldsymbol{g}&\boldsymbol{0}\\ \boldsymbol{0}&\boldsymbol{g}_v\end{pmatrix}\,,$$

$$\tilde{\boldsymbol{u}}(n)=[\,u(n)\;\;\eta(n)\,]^{\mathrm{T}}\,,$$

$$\tilde{\boldsymbol{c}}=\left[\boldsymbol{c}^{\mathrm{T}}\;\;\boldsymbol{c}_v^{\mathrm{T}}\right]^{\mathrm{T}}\,.$$

Then, the Kalman filter given in (43.190a) can easily be generalized to estimate both speech and noise.

43.6.3 Hidden-Markov-Model-Based Noise Reduction

The hidden Markov model (HMM) is a statistical model that uses a finite number of states and the associated

state transitions to jointly model the temporal and spectral variation of signals. It has long been used for speech modeling with applications to speech recognition [43.76] and noise reduction [43.36]. For noise reduction, the HMM-based method is basically similar to the statistical estimators described in Sect. 43.5. However, in Sect. 43.5, the estimators assume explicit knowledge of the joint probability distribution of the clean speech and noise signals, so that the conditional expected value of the clean speech (or its sample spectrum), given the noisy speech, can be evaluated, while in the HMM-based approach such statistical knowledge is not required to be known. Instead, the HMM method conquers the problem in two steps. In the first step, which is often called the training process, the probability distributions of the clean speech and the noise process are estimated from given training sequences. The estimated distributions are then applied in the second step to construct the desired speech estimators.

Typically, for a noise-reduction problem, we only require two ergodic HMMs: one for modeling the clean speech signal and the other for modeling the noise process. Consider HMMs with N states and M mixtures. Let $\boldsymbol{O} \triangleq \{\boldsymbol{o}_0, \boldsymbol{o}_1, \cdots, \boldsymbol{o}_t, \cdots, \boldsymbol{o}_T\}$ be the observed feature sequence, where $\boldsymbol{o}_t$ is a K-dimensional feature vector (e.g., consisting of K consecutive samples of the speech signal or the speech spectrum) and T is the duration of the sequence. Let $\boldsymbol{S} \triangleq \{s_0, s_1, \cdots, s_t, \cdots, s_T\}$, $s_t \in \{1, 2, \cdots, N\}$, be a sequence of states corresponding to $\boldsymbol{O}$. The pdf for $\boldsymbol{O}$ is written

$$p(\boldsymbol{O}) = \sum_{\boldsymbol{S}} p(\boldsymbol{O}, \boldsymbol{S}|\lambda) = \sum_{\boldsymbol{S}} \prod_{t=0}^{T} \alpha_{s_{t-1}s_t} b_{s_t}(\boldsymbol{o}_t) \,, \tag{43.195}$$

where $\alpha_{s_{t-1}s_t}$ is the transition probability from state s_{t-1} at time $t-1$ to state s_t at time t, $\alpha_{s_{-1}s_0} \triangleq \pi_{s_0}$ denotes the initial state probability, $b_{s_t}(\boldsymbol{o}_t)$ is the pdf of the observation vector, and λ denotes the HMM parameter set, i. e.,

$$\lambda \triangleq (\boldsymbol{\Pi}, \boldsymbol{A}, \boldsymbol{b}) \tag{43.196}$$

with

$$\boldsymbol{\Pi} = \{\pi_1, \pi_2, \cdots, \pi_N\} \tag{43.197}$$

being the set of initial state probabilities,

$$\boldsymbol{A} = \begin{pmatrix} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1N} \\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2N} \\ \vdots & \vdots & \cdots & \vdots \\ \alpha_{N1} & \alpha_{N2} & \cdots & \alpha_{\mathrm{NN}} \end{pmatrix} \tag{43.198}$$

being the set of state transition probabilities, and

$$\boldsymbol{b} = \{\, b_1(\boldsymbol{o}_t) \;\; b_2(\boldsymbol{o}_t) \;\; \cdots \;\; b_N(\boldsymbol{o}_t) \,\} \tag{43.199}$$

defining the pdf of the observed signal vector $\boldsymbol{O}$. For Gaussian-mixture autoregressive (AR) HMMs, the pdf $b_{s_t}(\boldsymbol{o}_t)$ is given by

$$b_{s_t}(\boldsymbol{o}_t) = \sum_{\kappa_t} c_{\kappa_t|s_t} b(\boldsymbol{o}_t|s_t, \kappa_t) \,, \tag{43.200}$$

where $\kappa_t \in [1, 2, \cdots, M]$ denotes the mixture component chosen at time t, $c_{\kappa_t|s_t}$ is the probability of choosing the mixture component κ_t given that the process is in state s_t, and $b(\boldsymbol{o}_t|s_t, \kappa_t)$ is the pdf of the observation vector $\boldsymbol{o}_t$ given κ_t and s_t. For a zero-mean Gaussian AR process with order P, we have

$$b(\boldsymbol{o}_t|s_t = n, \kappa_t = m) = \frac{\exp\left(-\frac{1}{2}\boldsymbol{o}_t^{\mathrm{T}} \boldsymbol{\Sigma}_{n,m}^{-1} \boldsymbol{o}_t\right)}{\sqrt{(2\pi)^K \det(\boldsymbol{\Sigma}_{n,m})}} \,, \tag{43.201}$$

where $\boldsymbol{\Sigma}_{n,m}$ is the covariance matrix, which is given by

$$\boldsymbol{\Sigma}_{n,m} = \sigma_{n,m}^2 \left(\boldsymbol{A}_{n,m}^H \boldsymbol{A}_{n,m}\right)^{-1} \,, \tag{43.202}$$

$\sigma_{n,m}^2$ is the variance of the AR process, and $\boldsymbol{A}_{n,m}$ is a $K \times K$ lower-triangular Toeplitz matrix with its first $P+1$ entries in the first column being the AR coefficients, i. e., $\boldsymbol{a}_{n,m} = [a_{n,m}(0), a_{n,m}(1), \cdots, a_{n,m}(P)]^{\mathrm{T}}$, and $a_{n,m}(0) = 1$ [43.34, 77]. In this case, the HMM parameter set λ consists of the following parameters: the initial state probabilities $\boldsymbol{\Pi}$, the state transition probabilities $\boldsymbol{A}$, and the AR parameters $\boldsymbol{a}_{n,m}$ and $\sigma_{n,m}^2$. Given a training data sequence, an ML estimate of λ (for both clean speech and noise) can be obtained using the Baum re-estimation algorithm [43.77]. Alternatively, the segmental k-means algorithm can be used to optimize the parameter set along the dominant state and mixture sequence [43.41].

Once the model parameters for both the clean speech and the noise signals are estimated, statistical estimators of clean speech can be constructed based on different criteria. For example, the MAP estimator can be formulated as

$$\hat{\boldsymbol{x}}_{\mathrm{MAP}} = \arg\max_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{s}_y, \lambda_x, \lambda_b) \,, \tag{43.203}$$

where $\boldsymbol{x} \triangleq \{\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_t\}$ denotes a sequence of the clean speech signal, $\boldsymbol{y} \triangleq \{\boldsymbol{y}_0, \boldsymbol{y}_1, \cdots, \boldsymbol{y}_\tau\}$ (with $\tau \geq t$) is the observation signal sequence, $\boldsymbol{s}_y$ denotes the composite state sequence of the noisy signal, and λ_x and λ_b are, respectively, the HMM model parameters for the clean speech and noise signals.

Since the composite state sequence $\boldsymbol{s}_y$ is not known, direct maximization of $p(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{s}_y, \lambda_x, \lambda_b)$ is not trivial. An alternative method is to achieve the maximization iteratively using the estimation-maximization (EM) algorithm [43.78]. Let i denote the iterative index (initially set to zero). The EM algorithm obtains a new speech estimate by maximizing the cost function

$$J[\boldsymbol{x}(i+1)] \triangleq \sum_{\boldsymbol{s}_y} \sum_{\boldsymbol{\kappa}} p\left[\boldsymbol{s}_y, \boldsymbol{\kappa}|\hat{\boldsymbol{x}}(i)\right] \ln p[\boldsymbol{s}_y, \boldsymbol{\kappa}, \boldsymbol{x}(i+1)|\boldsymbol{y}] \,, \tag{43.204}$$

where $\boldsymbol{\kappa} \triangleq \{\boldsymbol{\kappa}_0, \boldsymbol{\kappa}_1, \cdots, \boldsymbol{\kappa}_\tau\}$ is a sequence of mixture components corresponding to $\boldsymbol{y}$ and $\boldsymbol{s}_y$. The solution to the above optimization problem is

$$\hat{\boldsymbol{x}}_t(i+1) = \left[\sum_{n=1}^{N} \sum_{m=1}^{M} q_t[n, m|\hat{\boldsymbol{x}}(i)] H_{n,m}^{-1}\right]^{-1} \boldsymbol{y}_t \,, \tag{43.205}$$

where $q_t[n, m|\hat{\boldsymbol{x}}(i)] = P[q_t = n, \kappa_t = m|\hat{\boldsymbol{x}}(i)]$ is the conditional probability of being in state n and choosing mixture m at time frame t given an estimate of the clean speech, $\hat{\boldsymbol{x}}(i)$, and $H_{n,m}$ is a Wiener filter for the output Gaussian process from state n and mixture m, which can be constructed from the speech and noise covariance matrices given in (43.202) [43.34]. The iterative MAP estimator given in (43.205) can also be implemented in the frequency domain so as to exploit the efficiency of the FFT. In the frequency domain, the estimator can be written as

$$\hat{\boldsymbol{x}}_{t,\omega}(i+1) = \left[\sum_{n=1}^{N} \sum_{m=1}^{M} q_t[n, m|\hat{\boldsymbol{x}}(i)] H_{n,m}^{-1}(\omega)\right]^{-1} \boldsymbol{y}_{t,\omega} \,, \tag{43.206}$$

where $\hat{\boldsymbol{x}}_{t,\omega}(i+1)$ and $\boldsymbol{y}_{t,\omega}$ are the Fourier transforms of $\hat{\boldsymbol{x}}_t(i+1)$ and $\boldsymbol{y}_t$, respectively, and $H_{n,m}(\omega)$ is the frequency-domain counterpart of $H_{n,m}$, which can be constructed from the AR (LPC) coefficients of the speech and noise signals as in (43.185). Briefly, the iterative MAP estimator given in (43.205) and (43.206) can be summarized as follows.

1. The conditional probability $q_t[n, m|\hat{\boldsymbol{x}}(i)]$ [being in state n and mixture m at time t given an estimate of the clean speech, i. e., $\hat{\boldsymbol{x}}(i)$] is evaluated for all possible states n, mixture m, and time frames t (using the forward–backward algorithm [43.76]).
2. For each state and mixture pair (n, m), there is a set of AR (LPC) coefficients. This set of coefficients is used to form a Wiener filter $H_{n,m}^{-1}(\omega)$.
3. A new estimate of the clean speech is obtained by filtering the noisy speech through a weighted sum of Wiener filters (the weights are $q_t[n, m|\hat{\boldsymbol{x}}(i)]$).
4. The new speech estimate is used to re-estimate the conditional probability $q_t[n, m|\hat{\boldsymbol{x}}(i+1)]$.

These four steps are repeated until some preset convergence criterion is satisfied.

Similarly, we can formulate the MMSE estimator presented in [43.79]. Note that the HMM-based estimators rely solely on the statistics of the clean speech inferred from the training data. However, some speech properties, such as the time-varying gain (energy contour), cannot be reliably predicted in the training sequences and these properties have to be directly estimated from the noisy speech [43.36].

The advantage of the HMM-based estimators is that they do not need explicit knowledge of the speech and noise distributions. In addition, they can tolerate some nonstationarity of the noise signal, depending on the number of states and mixtures used in the noise HMM. However, distortion will arise when the characteristics of the noise are not represented in the training set.

43.7 Summary

Noise reduction has long been a major research focus of speech processing for communications. The fundamental objective of noise reduction is to mitigate or eliminate the effects of noise on the desired speech signal. The fulfillment of this objective, however, is not trivial. In general, the formulation of noise reduction varies from one application to another, and the difficulty and complexity of the problem depend on many factors such as the nature of the noise signal, the number of microphones available for noise reduction, and the nature of the speech communication system. This chapter was devoted to a particular yet ubiquitous problem, i. e., single-channel noise reduction. We formulated this particular problem as a signal/parameter estimation problem. Through theoretical analysis, we demonstrated, in terms of

noise attenuation and signal-to-noise ratio (SNR) improvement, that noise reduction is feasible in the single-microphone scenario. However, the price for this is speech distortion. In general, the more the noise reduction, the more the speech distortion. We reviewed the state of the art of noise-reduction techniques. The algorithms were classified into three broad categories, i. e., filtering techniques, spectral restoration, and speech-model-based methods. The basic principle and idea underlying each group of algorithms were outlined and their strengths and shortcomings were also discussed.

References

43.1 J. Benesty, S. Makino, J. Chen (Eds.): *Speech Enhancement* (Springer, Berlin, Heidelberg 2005)

43.2 D.H. Johnson, D.E. Dudgeon: *Array Signal Processing: Concepts and Techniques* (Prentice Hall, Upper Saddle River 1993)

43.3 M. Brandstein, D. Ward (Eds.): *Microphone Arrays: Signal Processing Techniques and Applications* (Springer, Berlin, Heidelberg 2001)

43.4 Y. Huang, J. Benesty (Eds.): *Audio Signal Processing for Next-Generation Multimedia Communication Systems* (Kluwer Academic, Boston 2004)

43.5 B. Widrow, J.R. Glover, J.M. McCool, J. Kaunitz, C.S. Williams, R.H. Hearn, J.R. Zeidler, E. Dong, R.C. Goodwin: Adaptive noise canceling: principles and applications, Proc. IEEE **63**, 1692–1716 (1975)

43.6 B. Widrow, S.D. Stearns: *Adaptive Signal Processing* (Prentice Hall, Englewood Cliffs 1985)

43.7 M.M. Goulding, J.S. Bird: Speech enhancement for mobile telephony, IEEE Trans. Veh. Technol. **39**, 316–326 (1990)

43.8 H.J. Kushner: On closed-loop adaptive noise cancellation, IEEE Trans. Automat. Contr. **43**, 1103–1107 (1998)

43.9 A.S. Abutaled: An adaptive filter for noise canceling, IEEE Trans. Circuits Syst. **35**, 1201–1209 (1998)

43.10 M. R. Schroeder: U.S. Patent No. 3180936, filed Dec. 1, 1960, issued Apr. 27, 1965

43.11 M. R. Schroeder: U.S. Patent No. 3403224, filed May 28, 1965, issued Sept. 24, 1968

43.12 S.F. Boll: Suppression of acoustic noise in speech using spectral subtraction, IEEE Trans. Acoust. Speech Signal Process. **ASSP-27**, 113–120 (1979)

43.13 J.S. Lim, A.V. Oppenheim: Enhancement and bandwidth compression of noisy speech, Proc. IEEE **67**, 1586–1604 (1979)

43.14 J.S. Lim (Ed.): *Speech Enhancement* (Prentice Hall, Englewood Cliffs 1983)

43.15 P. Vary: Noise suppression by spectral magnitude estimation–mechanism and theoretical limits, Signal Process. **8**, 387–400 (1985)

43.16 R. Martin: Noise power spectral density estimation based on optimal smoothing and minimum statistics, IEEE Trans. Speech Audio Process. **9**, 504–512 (2001)

43.17 W. Etter, G.S. Moschytz: Noise reduction by noise-adaptive spectral magnitude expansion, J. Audio Eng. Soc. **42**, 341–349 (1994)

43.18 J. Chen, J. Benesty, Y. Huang, S. Doclo: New insights into the noise reduction Wiener filter, IEEE Trans. Speech Audio Process. **14**, 1218–1234 (2006)

43.19 Y. Ephraim, H.L. Van Trees: A signal subspace approach for speech enhancement, IEEE Trans. Speech Audio Process. **3**, 251–266 (1995)

43.20 M. Dendrinos, S. Bakamidis, G. Garayannis: Speech enhancement from noise: A regenerative approach, Speech Commun. **10**, 45–57 (1991)

43.21 P.S.K. Hansen: *Signal Subspace Methods for Speech Enhancement*, Ph.D. Dissertation (Tech. Univ. Denmark, Lyngby 1997)

43.22 S.H. Jensen, P.C. Hansen, S.D. Hansen, J.A. Sørensen: Reduction of broad-band noise in speech by truncated QSVD, IEEE Trans. Speech Audio Process. **3**, 439–448 (1995)

43.23 H. Lev-Ari, Y. Ephraim: Extension of the signal subspace speech enhancement approach to colored noise, IEEE Trans. Speech Audio Process. **10**, 104–106 (2003)

43.24 A. Rezayee, S. Gazor: An adaptive KLT approach for speech enhancement, IEEE Trans. Speech Audio Process. **9**, 87–95 (2001)

43.25 U. Mittal, N. Phamdo: Signal/noise KLT based approach for enhancing speech degraded by colored noise, IEEE Trans. Speech Audio Process. **8**, 159–167 (2000)

43.26 Y. Hu, P.C. Loizou: A generalized subspace approach for enhancing spech corrupted by colored noise, IEEE Trans. Speech Audio Process. **11**, 334–341 (2003)

43.27 Y. Ephraim, D. Malah: Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator, IEEE Trans. Acoust. Speech Signal Process. **32**, 1109–1121 (1984)

43.28 Y. Ephraim, D. Malah: Speech enhancement using a minimum mean-square error log-spectral amplitude estimator, IEEE Trans. Acoust. Speech Signal Process. **ASSP-33**, 443–445 (1985)

43.29 R.J. McAulay, M.L. Malpass: Speech enhancement using a soft-decision noise suppression filter, IEEE Trans. Acoust. Speech Signal Process. **28**, 137–145 (1980)

43.30 P.J. Wolfe, S.J. Godsill: Simple alternatives to the Ephraim and Malah suppression rule for speech ehancemnet, Proc. IEEE ICASSP **2001**, 496–499 (2001)

43.31 K.K. Paliwal, A. Basu: A speech enhancement method based on Kalman filtering, Proc. IEEE ICASSP **1987**, 177–180 (1987)

43.32 J.D. Gibson, B. Koo, S.D. Gray: Filtering of colored noise for speech enhancement and coding, IEEE Trans. Signal Process. **39**, 1732–1742 (1991)

43.33 S. Gannot, D. Burshtein, E. Weinstein: Iterative and sequential Kalman filter-based speech enhancement algorithms, IEEE Trans. Speech Audio Process. **6**, 373–385 (1998)

43.34 Y. Ephraim, D. Malah, B.-H. Juang: On the application of hidden Markov models for enhancing noisy speech, IEEE Trans. Acoust. Speech Signal Process. **37**, 1846–1856 (1989)

43.35 Y. Ephraim: A Bayesian estimation approach for speech enhancement using hidden Markov models, IEEE Trans. Signal Process. **40**, 725–735 (1992)

43.36 Y. Ephraim: Statstical-model-based speech enhancement systems, Proc. IEEE **80**, 1526–1555 (1992)

43.37 D. Klatt: Review of test-to-speech conversion for English, J. Acoust. Soc. Am. **82**, 737–793 (1987)

43.38 U. Jekosch: Speech quality assessment and evaluation, Proc. Eurospeech **1993**, 1387–1394 (1993)

43.39 C. Delogu, P. Paoloni, P. Pocci, C. Sementina: Quality evaluation of text-to-speech synthesizers using magnitude estimation, categorical estimation, pair comparison and reaction time methods, Proc. Eurospeech **1991**, 353–356 (1991)

43.40 S.R. Quackenbush, T.P. Barnwell, M.A. Clements: *Objective Measures of Speech Quality* (Prentice Hall, Englewood Cliffs 1988)

43.41 L.R. Rabiner, B.H. Juang: *Fundamentals of Speech Recognition* (Prentice Hall, Englewood Cliffs 1993)

43.42 D. Mansour, B.H. Juang: A family of distortion meansures based upon projection operation for robust speech recognition, IEEE Trans. Acoust. Speech Signal Process. **37**, 1659–1671 (1989)

43.43 F. Itakura, S. Saito: A statistical method for estimation of speech spectral density and formant frequencies, Electron. Commun. Jpn. **53A**, 36–43 (1970)

43.44 G. Chen, S.N. Koh, I.Y. Soon: Enhanced Itakura measure incorporating masking properties of human auditory system, Signal Process. **83**, 1445–1456 (2003)

43.45 K. Fukunaga: *Introduction to Statistial Pattern Recognition* (Academic, San Diego 1990)

43.46 N. Wiener: *Extrapolation, Interpolation, and Smoothing of Stationary Time Series* (Wiley, New York 1949)

43.47 H.L. Van Trees: *Dection, Estimation, and Modulation Theory*, Part I (Wiley, New York 1968)

43.48 R. Martin: Speech enhancement using MMSE short time spectral estimation with Gamma distributed speech priors, Proc. IEEE ICASSP **2002**, I253–I256 (2002)

43.49 I.S. Gradshteyn, I.M. Ryzhik, A. Jeffery, D. Zwillinger (Eds.): *Table of Integrals, Series, and Products* (Academic, San Diego 2000)

43.50 C. Breithaupt, R. Martin: MMSE estimation fo magnitude-square DFT coefficients with supergaussian priors, Proc. IEEE ICASSP **2003**, I848–I851 (2003)

43.51 I. Cohen: Speech enhancement using supergaussian speech models and noncausal a priori SNR estimation, Speech Commun. **47**, 336–350 (2005)

43.52 S.O. Rice: Stasitical properties of a sinewave plus random noise, Bell System Tech. J. **0**, 109–157 (1948)

43.53 D. Middleton, R. Esposito: Simultaneous optimum detection and estimation of signals in noise, IEEE Trans. Inform. Theory **IT-14**, 434–444 (1968)

43.54 D.L. Wang, J.S. Lim: The unimportance of phase in speech enhancement, IEEE Trans. Acoust. Speech Signal Process. **ASSP-30**, 679–681 (1982)

43.55 H. Dudley, T.H. Tarnoczy: The speaking machine of Wolfgang von Kempelen, J. Acoust. Soc. Am. **22**, 151–166 (1950)

43.56 Sir R. Paget: *Human Speech* (Harcourt, London, New York 1930)

43.57 J.Q. Stewart: An electrical analogue of the vocal cords, Nature **110**, 311–312 (1922)

43.58 H.K. Dunn: The calculation of vowel resonances, and an electrical vocal tract, J. Acoust. Soc. Am. **22**, 740–753 (1950)

43.59 B.S. Atal, L.S. Hanauer: Speech analysis and synthesis by linear prediction of the speech wave, J. Acoust. Soc. Am. **50**, 637–655 (1971)

43.60 F. Itakura: Minimum prediction residual principle applied to speech recognition, IEEE Trans. Acoust. Speech Signal Process. **ASSP-23**, 67–72 (1975)

43.61 T.W. Parsons: Separation of speech from interfering speech by means of harmonic selection, J. Acoust. Soc. Am. **60**, 911–918 (1976)

43.62 R.H. Frazier, S. Samsam, L.D. Braida, A.V. Oppenheim: Enhancement of speech by adaptive filtering, Proc. IEEE ICASSP **1976**, 251–253 (1976)

43.63 R.J. McAulay, T.F. Quatieri: Mid-rate coding based on sinusoidal representation of speech, Proc IEEE ICASSP **1985**, 945–948 (1985)

43.64 D.P. Morgan, E.B. George, L.T. Lee, S.M. Kay: Cochannel speaker separation by harmonic enhancement and suprresion, IEEE Trans. Speech Audio Process. **5**, 405–424 (1997)

43.65 E.B. George, M.J.T. Smith: Speech analysis/synthesis and modification using an analysis-by-synthesis/overlap-add sinusoidal model, IEEE Trans. Speech Audio Process. **5**, 389–406 (1997)

43.66 D. O'Brien, A.I.C. Monaghan: Concatenative synthesis based on a harmonic model, IEEE Trans. Speech Audio Process. **9**, 11–20 (2001)

43.67 J.S. Lim, A.V. Oppenheim, L.D. Braida: Evaluation of an adpative comb filtering method for enhancing speech degraded by white noise addition,

IEEE Trans. Acoust. Speech Signal Process. **ASSP-26**, 354–358 (1978)

43.68 J. Makhoul: Linear prediction: A tutorial review, Proc. IEEE **63**, 561–580 (1975)

43.69 J.R. Deller, J.G. Proakis, J.H.L. Hansen: *Discrete-Time Processing of Speech Signals* (Macmillan, New York 1993)

43.70 K.M. Malladi, R.V. Rajakumar: Estimation of time-varying AR models of speech through Gauss-Markov modeling, Proc. IEEE ICASSP **6**, 305–308 (2003)

43.71 M. Niedźwiecki, K. Cisowski: Adaptive scheme for elimination fo broadband noise and impulsive disturbance from AR and ARMA signals, IEEE Trans. Signal Process. **44**, 528–537 (1996)

43.72 B. Koo, J.D. Gibson: Filtering of colored noise for speech enhancement and coding, Proc. IEEE ICASSP **1989**, 345–352 (1989)

43.73 B. Lee, K.Y. Lee, S. Ann: An EM-based approach for parameter enhancement with an application to speech signals, Signal Process. **46**, 1–14 (1995)

43.74 Z. Goh, K.C. Tan, B.T.G. Tan: Kalman-filtering speech enhancement method based on a voiced-unvoiced speech model, IEEE Trans. Speech Audio Process. **7**, 510–524 (1999)

43.75 C. Li, S.V. Andersen: Intergrating Kalman filtering and multi-pulse coding for speech enhancement with a non-stationary model of the speech signal, Proc. IEEE ICASSP **2004**, 2300–2304 (2004)

43.76 L.R. Rabiner: A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE **77**, 257–286 (1989)

43.77 B.H. Juang, L.R. Rabiner: Mixture autoregressive hidden Markov models for speech signals, IEEE Trans. Acoust. Speech Signal Process. **ASSP-33**, 1404–1413 (1985)

43.78 A.P. Dempster, N.M. Laird, D.B. Rubin: Maximum likelihood from incomplete data via the EM algorithm, J. Roy. Stat. Soc. B **39**, 1–38 (1977)

43.79 H. Sameti, H. Sheikhzadeh, L. Deng, R.L. Brennan: HMM-based strategies for enhancement of speech signals embedded in nonstationary noise, IEEE Trans. Speech Audio Process. **6**, 445–455 (1998)

44. Spectral Enhancement Methods

I. Cohen, S. Gannot

In this chapter, we focus on the statistical methods that constitute a speech spectral enhancement system and describe some of their fundamental components. We begin in Sect. 44.2 by formulating the problem of spectral enhancement. In Sect. 44.3, we address the time–frequency correlation of spectral coefficients for speech and noise signals, and present statistical models that conform with these characteristics. In Sect. 44.4, we present estimators for speech spectral coefficients under speech presence uncertainty based on various fidelity criteria. In Sect. 44.5, we address the problem of speech presence probability estimation. In Sect. 44.6, we present useful estimators for the a priori signal-to-noise ratio (SNR) under speech presence uncertainty. We present the decision-directed approach, which is heuristically motivated, and the recursive estimation approach, which is based on statistical models and follows the rationale of Kalman filtering. In Sect. 44.7, we describe the improved minima-controlled recursive averaging (IMCRA) approach for noise power spectrum estimation. In Sect. 44.8, we provide a detailed example of a speech enhancement algorithm, and demonstrate its performance in environments with various noise types. In Sect. 44.9, we survey the main types of spectral enhancement components, and discuss the significance of the choice of statistical model, fidelity criterion, a priori SNR estimator, and noise spectrum estimator. Some concluding comments are made in Sect. 44.10.

The problem of spectral enhancement of noisy speech signals from a single microphone has attracted considerable research effort for over 30 years. It is a problem with numerous applications ranging from speech recognition, to hearing aids and hands-free mobile communication. In this chapter, we present the fundamental components that constitute a speech spectral enhancement system. We describe statistical models that take into consideration the time correlation between successive spectral components of the speech signal, and present estimators for the speech spectral coefficients based on various fidelity criteria. We address the problem of a priori SNR estimation under speech presence uncertainty, and noise power spectrum estimation. We also provide a detailed design example of a speech enhancement algorithm.

44.1 Spectral Enhancement

Spectral enhancement of noisy speech has been a challenging problem for many researchers for over 30 years, and is still an active research area (see e.g., [44.1–3] and references therein). This problem is often formulated as the estimation of speech spectral components from a speech signal degraded by statistically independent additive noise. In this chapter we consider spectral enhancement methods for single-channel set-ups, assuming that only one-microphone noisy output is available for the estimation. The situation of one-microphone setups is particularly difficult under nonstationary noise and a low signal-to-noise ratio (SNR), since no reference signal is available for the estimation of the background noise.

A variety of different approaches for spectral enhancement of noisy speech signals have been introduced over the years. One of the earlier methods, and perhaps the most well-known approach, is spectral subtraction [44.4, 5], in which an estimate of the short-term power spectral density of the clean signal is obtained by subtracting an estimate of the power spectral density of the background noise from the short-term power spectral density of the degraded signal. The square root of the resulting estimate is considered an estimate of the spectral magnitude of the speech signal. Subsequently, an estimate of the signal is obtained by combining the spectral magnitude estimate with the complex exponential of the phase of the noisy signal. This method generally results in random narrowband fluctuations in the residual noise, also known as musical tones, which is annoying and disturbing to the perception of the enhanced signal. Many variations have been developed to cope with the musical residual noise phenomena [44.4, 6–9], including spectral subtraction techniques based on masking properties of the human auditory system [44.10, 11].

The spectral subtraction method makes minimal assumptions about the signal and noise, and when carefully implemented, produces enhanced signals that may be acceptable for certain applications. Statistical methods [44.12–16] are designed to minimize the expected value of some distortion measure between the clean and estimated signals. This approach requires the presumption of reliable statistical models for the speech and noise signals, the specification of a perceptually meaningful distortion measure, and a mathematically tractable derivation of an efficient signal estimator. A statistical speech model and perceptually meaningful distortion measure, which are the most appropriate for spectral enhancement, have not yet been determined. Hence, statistical methods for spectral enhancement mainly differ in their statistical model [44.12, 14, 15], distortion measure [44.17–19], and the particular implementation of the spectral enhancement algorithm [44.2].

Spectral enhancement based on hidden Markov processes (HMPs) try to circumvent the assumption of specific distributions for the speech and noise processes [44.20–23]. The probability distributions of the two processes are first estimated from long training sequences of clean speech and noise samples, and then used jointly with a given distortion measure to derive an estimator for the speech signal. Normally, vectors generated from a given sequence of states are assumed to be statistically independent. However, the HMP can be extended to take into account the time–frequency correlation of speech signals by using nondiagonal covariance matrices for each subsource, and by assuming that a sequence of vectors generated from a given sequence of states is a nonzero-order autoregressive process [44.21, 24]. HMP-based speech enhancement relies on the type of training data [44.25]. It works best with the type of noise used during training, and often worse with other types of noise. Furthermore, improved performance generally entails more-complex models and greater computational requirements. While hidden Markov models have been successfully applied to automatic recognition of clean speech signals [44.26, 27], they were not found to be sufficiently refined models for speech enhancement applications [44.3].

Subspace methods [44.28–31] attempt to decompose the vector space of the noisy signal into a signal-plus-noise subspace and a noise subspace. Spectral enhancement is performed by removing the noise subspace and estimating the speech signal from the remaining subspace. The signal subspace decomposition can be achieved by either using the Karhunen–Loève transform (KLT) via eigenvalue decomposition of a Toeplitz covariance estimate of the noisy vector [44.28, 30], or by using the singular value decomposition of a data matrix [44.32, 33]. Linear estimation in the signal-plus-noise subspace is performed with the goal of minimizing signal distortion while masking the residual noise by the signal. A perceptually motivated signal subspace approach takes into account the masking properties of the human auditory system and reduces the perceptual effect of the residual noise [44.34, 35].

44.2 Problem Formulation

Let $x(n)$ and $d(n)$ denote speech and uncorrelated additive noise signals, respectively, where n is a discrete-time index. The observed signal $y(n)$, given by $y(n) = x(n) + d(n)$, is transformed into the time-frequency domain by applying the short-time Fourier transform (STFT). Specifically,

$$Y_{tk} = \sum_{n=0}^{N-1} y(n+tM)h(n)\mathrm{e}^{-\mathrm{i}\frac{2\pi}{N}nk}\,, \tag{44.1}$$

where t is the time frame index ($t = 0, 1, \ldots, k$) is the frequency bin index ($k = 0, 1, \ldots, N-1$), $h(n)$ is an analysis window of size N (e.g., Hamming window), and M is the framing step (number of samples separating two successive frames). Given an estimate $\hat{X}_{tk}$ for the STFT of the clean speech (Fig. 44.1), an estimate for the clean speech signal is obtained by applying the inverse STFT,

$$\hat{x}(n) = \sum_{t}\sum_{k=0}^{N-1} \hat{X}_{tk}\tilde{h}(n-tM)\mathrm{e}^{\mathrm{i}\frac{2\pi}{N}k(n-tM)}\,, \tag{44.2}$$

where $\tilde{h}(n)$ is a synthesis window that is biorthogonal to the analysis window $h(n)$ [44.36], and the inverse STFT is efficiently implemented by using the weighted overlap-add method [44.37] (see also Sect. 44.8).

The spectral enhancement problem is generally formulated as deriving an estimator $\hat{X}_{tk}$ for the speech spectral coefficients, such that the expected value of a certain distortion measure is minimized. Let $d(X_{tk}, \hat{X}_{tk})$ denote a distortion measure between X_{tk} and its estimate $\hat{X}_{tk}$, and let ψ_t represent the information set that can be employed for the estimation at frame t (e.g., the noisy data observed through time t). Let H_1^{tk} and H_0^{tk} denote, respectively, hypotheses of signal presence and absence in the noisy spectral coefficient Y_{tk}:

$$\begin{aligned} H_1^{tk}: &\quad Y_{tk} = X_{tk} + D_{tk} \\ H_0^{tk}: &\quad Y_{tk} = D_{tk}\,. \end{aligned}$$

Let $\hat{p}_{tk} = P(H_1^{tk}|\psi_t)$ denote an estimate for the signal presence probability, $\hat{\lambda}_{tk} = E\{|X_{tk}|^2|H_1^{tk}, \psi_t\}$ denote an estimate for the variance of a speech spectral coefficient X_{tk} under H_1^{tk}, and $\widehat{\sigma_{tk}^2} = E\{|Y_{tk}|^2|H_0^{tk}, \psi_t\}$ denote an estimate for the variance of a noise spectral coefficient D_{tk}. Then, we consider an estimator for X_{tk} which minimizes the expected distortion given $\hat{p}_{tk}$, $\hat{\lambda}_{tk}$, $\widehat{\sigma_{tk}^2}$ and the noisy spectral coefficient Y_{tk}:

$$\min_{\hat{X}_{tk}} E\left\{d\left(X_{tk}, \hat{X}_{tk}\right)\middle|\hat{p}_{tk}, \hat{\lambda}_{tk}, \widehat{\sigma_{tk}^2}, Y_{tk}\right\}\,. \tag{44.3}$$

In particular, restricting ourselves to a squared error distortion measure of the form

$$d\left(X_{tk}, \hat{X}_{tk}\right) = \left|g(\hat{X}_{tk}) - \tilde{g}(X_{tk})\right|^2\,, \tag{44.4}$$

where $g(X)$ and $\tilde{g}(X)$ are specific functions of X (e.g., X, $|X|$, $\log|X|$, $\mathrm{e}^{\mathrm{i}\angle X}$), the estimator $\hat{X}_{tk}$ is calculated from

$$\begin{aligned} g(\hat{X}_{tk}) &= E\left\{\tilde{g}(X_{tk})\middle|\hat{p}_{tk}, \hat{\lambda}_{tk}, \widehat{\sigma_{tk}^2}, Y_{tk}\right\} \\ &= \hat{p}_{tk}E\left\{\tilde{g}(X_{tk})\middle|H_1^{tk}, \hat{\lambda}_{tk}, \widehat{\sigma_{tk}^2}, Y_{tk}\right\} \\ &\quad + (1-\hat{p}_{tk})E\left\{\tilde{g}(X_{tk})\middle|H_0^{tk}, Y_{tk}\right\}\,. \end{aligned} \tag{44.5}$$

Hence, the design of a particular estimator for X_{tk} requires the following specifications:

- Functions $g(X)$ and $\tilde{g}(X)$, which determine the fidelity criterion of the estimator
- A conditional probability density function (pdf) $p(X_{tk}|\lambda_{tk}, H_1^{tk})$ for X_{tk} under H_1^{tk} given its variance λ_{tk}, which determines the statistical model
- An estimator $\hat{\lambda}_{tk}$ for the speech spectral variance
- An estimator $\widehat{\sigma_{tk}^2}$ for the noise spectral variance
- An estimator $\hat{p}_{tk|t-1} = P(H_1^{tk}|\psi_{t-1})$ for the a priori signal presence probability, where ψ_{t-1} represents the information set known prior to having the measurement Y_{tk}

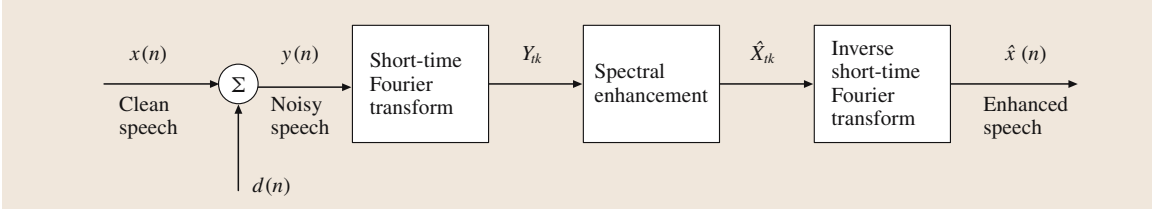


Fig. 44.1 Spectral enhancement approach

Given the a priori signal presence probability $\hat{p}_{tk|t-1}$, the (a posteriori) signal presence probability can be obtained from Bayes' rule:

$$\hat{p}_{tk} = P\left(H_1^{tk}\middle|Y_{tk}, \psi_{t-1}\right) = \left[1 + \frac{\left(1-\hat{p}_{tk|t-1}\right)p\left(Y_{tk}|H_0^{tk}, \psi_{t-1}\right)}{\hat{p}_{tk|t-1}\,p\left(Y_{tk}|H_1^{tk}, \psi_{t-1}\right)}\right]^{-1} . \quad (44.6)$$

In the following sections we present statistical models for speech signals in the STFT domain, and address the estimation problem of the speech spectral coefficient X_{tk} given $\hat{\lambda}_{tk}$, $\widehat{\sigma_{tk}^2}$, and $\hat{p}_{tk}$. Then we consider the estimation of the speech spectral variance λ_{tk}, the noise spectral variance σ_{tk}^2, and the speech presence probability $P(H_1^{tk})$, and describe an example of a speech enhancement algorithm.

44.3 Statistical Models

In this section, we present statistical models that take into account the time correlation between successive spectral components of the speech signal. To see graphically the relation between successive spectral components of a speech signal, in comparison with a noise signal, we present scatter plots for successive spectral magnitudes, and investigate the sample autocorrelation coefficient sequences (ACSs) of the STFT coefficients along time trajectories (the frequency bin index k is held fixed). We consider a speech signal that is constructed from six different utterances, without intervening pauses. The utterances, half from male speakers and half from female speakers, are taken from the TIMIT database [44.38]. (A corpus of phonemically and lexically transcribed speech of American English speakers of different sexes and dialects. The speech was recorded at Texas Instruments (TI) and transcribed at Massachusetts Institute of Technology (MIT), hence the corpus' name.) The speech signal is sampled at 16 kHz, and transformed into the STFT domain using Hamming analysis windows of 512 samples (32 ms) length, and 256 samples framing step (50% overlap between successive frames).

Figure 44.2 shows an example of scatter plots for successive spectral magnitudes of white Gaussian noise (WGN) and speech signals. It implies that 50% overlap between successive frames does not yield a significant correlation between the spectral magnitudes of the WGN signal. However, successive spectral magnitudes of the speech signal are highly correlated. Figure 44.3 shows the ACSs of the speech spectral components along time trajectories, for various frequency bins and framing steps. The 95% confidence limits [44.39] are depicted as horizontal dotted lines. In order to prevent an upward bias of the autocovariance estimates due to irrelevant (nonspeech) spectral components, the ACSs are computed from spectral components whose magnitudes are within 30 dB of the maximal magnitude. Specifically, the sample autocorrelation coefficients of the spectral magnitudes are calculated by

$$\rho_m = \frac{\sum_{t\in\mathcal{T}}\left(A_{tk}-\overline{A_k}\right)\left(A_{t+m,k}-\overline{A_k}\right)}{\sum_{t\in\mathcal{T}}\left(A_{tk}-\overline{A_k}\right)^2} , \quad (44.7)$$

where $A_{tk} \triangleq |X_{tk}|$ denotes the magnitude of X_{tk},

$$\overline{A_k} = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} A_{tk}$$

denotes the sample mean, m is the lag in frames, and $\mathcal{T}$ represents the set of relevant spectral components

$$\mathcal{T} = \left\{t \,\middle|\, A_{tk} \geq 10^{-30/20}\max_t\{A_{tk}\}\right\} .$$

The corresponding sample autocorrelation coefficients of the spectral phases are obtained by

$$\varrho_m = \frac{\sum_{t\in\mathcal{T}}\varphi_{tk}\varphi_{t+m,k}}{\sum_{t\in\mathcal{T}}\varphi_{tk}^2} , \quad (44.8)$$

where φ_{tk} denotes the phase of X_{tk}. Figure 44.4 shows the variation of the correlation between successive spectral magnitudes on frequency and on overlap between successive frames. Figures 44.3 and 44.4 demonstrate that, for speech signals, successive spectral magnitudes are highly correlated, while the correlation is generally larger at lower frequencies, and increases as the overlap between successive frames increases. As a comparison, the variation of ρ_1 on the overlap between frames is also shown for a realization of WGN (Fig. 44.4b, dotted line). It implies that, for a sufficiently large framing step ($M \geq N/2$, i. e., overlap

between frames $\leq 50\%$), successive spectral components of the *noise* signal, but clearly not of the *speech* signal, can be assumed uncorrelated. For smaller framing steps, the correlation between successive spectral noise components also has to be taken into consideration. Furthermore, since the length of the analysis window cannot be too large (its typical length is 20–40 ms [44.12]), adjacent Fourier expansion coefficients of the noise signal, D_{tk} and $D_{t,k+1}$, as well as adjacent coefficients of the speech signal, X_{tk} and $X_{t,k+1}$, are also correlated to a certain degree. Nevertheless, it is commonly assumed that expansion coefficients in different frequency bins are statistically independent [44.12, 15, 16, 40]. This allows one to formulate independent estimation problems for each frequency bin k, which greatly simplifies the resulting algorithms. In view of this discussion, we employ statistical models in the STFT domain that rely on the following set of assumptions [44.41].

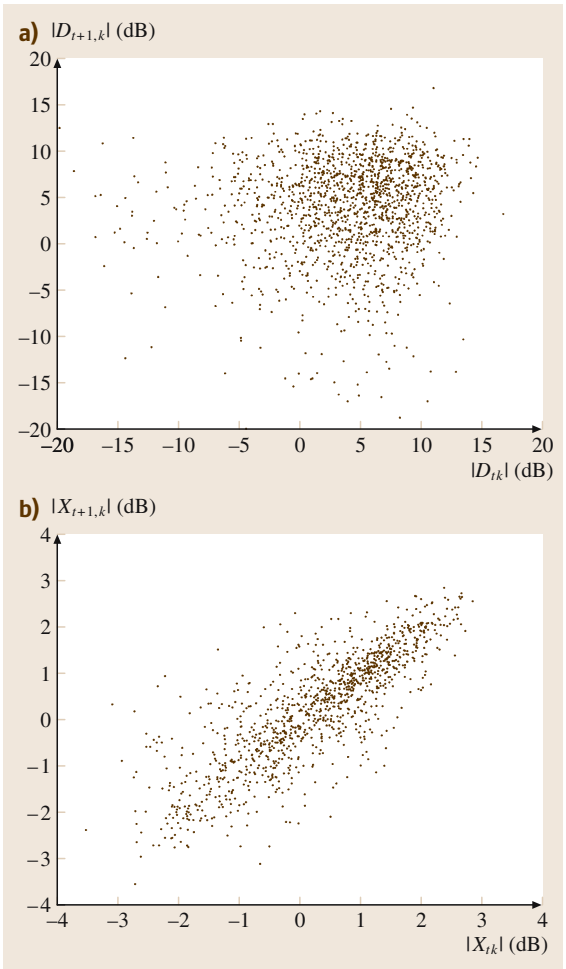


Fig. 44.2a,b Scatter plots for successive spectral magnitudes of **(a)** a white Gaussian noise signal, and **(b)** a speech signal at a center frequency of 500 Hz ($k = 17$). The overlap between successive frames is 50% (after [44.15])

1. The noise spectral coefficients $\{D_{tk}\}$ are zero-mean statistically independent Gaussian random variables. The real and imaginary parts of D_{tk} are independent and identically distributed (iid) random variables $\mathcal{N}(0, \sigma_{tk}^2/2)$.
2. Given $\{\lambda_{tk}\}$ and the state of speech presence in each time–frequency bin (H_1^{tk} or H_0^{tk}), the speech spectral coefficients $\{X_{tk}\}$ are generated by

$$X_{tk} = \sqrt{\lambda_{tk}}\, V_{tk} \,, \tag{44.9}$$

where $\{V_{tk}|H_0^{tk}\}$ are identically zero, and $\{V_{tk}|H_1^{tk}\}$ are statistically independent complex random variables with zero mean, unit variance, and iid real and imaginary parts:

$$\begin{aligned} H_1^{tk}: &\quad E\{V_{tk}\} = 0,\, E\left\{|V_{tk}|^2\right\} = 1 \,, \\ H_0^{tk}: &\quad V_{tk} = 0 \,. \end{aligned} \tag{44.10}$$

3. The pdf of V_{tk} under H_1^{tk} is determined by the specific statistical model. Let $V_{\mathrm{R}tk} = \mathrm{Re}\{V_{tk}\}$ and $V_{\mathrm{I}tk} = \mathrm{Im}\{V_{tk}\}$ denote, respectively, the real and imaginary parts of V_{tk}. Let $p(V_{\rho tk}|H_1^{tk})$ denote the pdf of $V_{\rho tk}$ ($\rho \in \{\mathrm{R}, \mathrm{I}\}$) under H_1^{tk}. Then, for a Gaussian model

$$p\left(V_{\rho tk}|H_1^{tk}\right) = \frac{1}{\sqrt{\pi}} \exp\left(-V_{\rho tk}^2\right) , \tag{44.11}$$

for a gamma model

$$\begin{aligned} &p\left(V_{\rho tk}|H_1^{tk}\right) \\ &= \frac{\sqrt[4]{3}}{2\sqrt{\pi}\sqrt[4]{2}} |V_{\rho tk}|^{-1/2} \exp\left(-\frac{\sqrt{3}|V_{\rho tk}|}{\sqrt{2}}\right) , \end{aligned} \tag{44.12}$$

and for a Laplacian model

$$p\left(V_{\rho tk}|H_1^{tk}\right) = \exp(-2|V_{\rho tk}|) \,. \tag{44.13}$$

4. The sequence of speech spectral variances $\{\lambda_{tk}|t = 0, 1, \ldots\}$ is a random process, which is generally correlated with the sequence of speech spectral magnitudes $\{A_{tk}|t = 0, 1, \ldots\}$. However, given λ_{tk}, A_{tk} is statistically independent of $A_{t'k'}$ for all $t \neq t'$ and $k \neq k'$.

Clearly, the first assumption does not hold when the overlap between successive frames is too large (Fig. 44.4b,

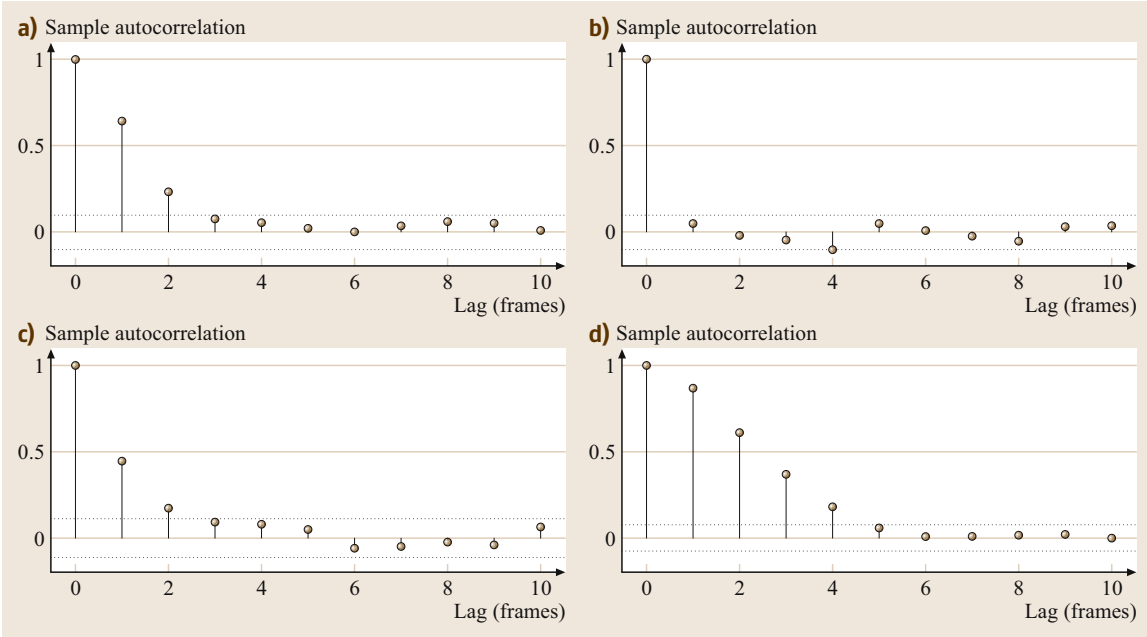
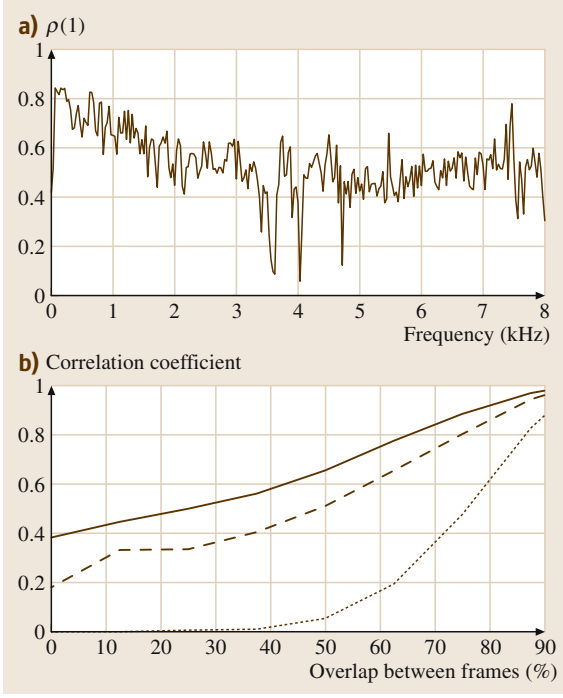


Fig. 44.3a–d Sample autocorrelation coefficient sequences (ACSs) of clean-speech STFT coefficients along time trajectories, for various frequency bins and framing steps. The *dotted lines* represent 95% confidence limits. **(a)** ACS of the spectral magnitude at frequency bin $k = 17$ (center frequency 500 Hz), framing step $M = N/2$ (50% overlap between frames). **(b)** ACS of the spectral phase, $k = 17$, $M = N/2$. **(c)** ACS of the spectral magnitude, $k = 65$ (center frequency 2 kHz), $M = N/2$. **(d)** ACS of the spectral magnitude, $k = 17$, $M = N/4$ (75% overlap between frames) (after [44.15])



implemented in accordance with this assumption (e.g., the overlap is not greater than 50%). The second assumption implies that the speech spectral coefficients $\{X_{tk}|H_1^{tk}\}$ are conditionally zero-mean statistically independent random variables given their variances $\{\lambda_{tk}\}$. The real and imaginary parts of X_t under H_1^t are conditionally iid random variables given λ_{tk}, satisfying

$$p\left(X_{\rho tk}|\lambda_{tk}, H_1^{tk}\right) = \frac{1}{\sqrt{\lambda_{tk}}} p\left(V_{\rho tk} = \frac{X_{\rho tk}}{\sqrt{\lambda_{tk}}}\middle| H_1^{tk}\right), \quad (44.14)$$

where $\rho \in \{\mathrm{R}, \mathrm{I}\}$. The last assumption allows one to take into account the time correlation between

Fig. 44.4a,b Variation of the correlation coefficient between successive spectral magnitudes. **(a)** Typical variation of ρ_1 with frequency for a speech signal and 50% overlap between frames. **(b)** Typical variation of ρ_1 with overlap between frames for a speech signal at center frequencies of 1 kHz (*solid line*) and 2 kHz (*dashed line*), and for a realization of white Gaussian noise (*dotted line*) (after [44.15])

successive spectral coefficients of the speech signal, while still considering the scalar estimation problem formulated in (44.3). Note that successive spectral coefficients are correlated, since the random processes $\{X_{tk}|t=0,1,\dots\}$ and $\{\lambda_{tk}|t=0,1,\dots\}$ are not independent.

44.4 Signal Estimation

In this section, we derive estimators for X_{tk} using various fidelity criteria, assuming that $\hat{p}_{tk}$, $\hat{\lambda}_{tk}$, and $\widehat{\sigma_{tk}^2}$ are given. Fidelity criteria that are of particular interest for speech enhancement applications are minimum mean-squared error (MMSE) [44.5], MMSE of the spectral amplitude (MMSE-SA) [44.12], and MMSE of the log-spectral amplitude (MMSE-LSA) [44.17,42]. The MMSE estimator is derived by substituting into (44.5) the functions

$$g(\hat{X}_{tk}) = \hat{X}_{tk}$$
$$\tilde{g}(X_{tk}) = \begin{cases} X_{tk}\,, & \text{under } H_1^{tk} \\ G_{\min} Y_{tk}\,, & \text{under } H_0^{tk}\,, \end{cases} \tag{44.15}$$

where $G_{\min} \ll 1$ represents a constant attenuation factor, which retains the noise naturalness during speech absence [44.2,42].

The MMSE-SA estimator is obtained by using the functions

$$g(\hat{X}_{tk}) = |\hat{X}_{tk}|\,,$$
$$\tilde{g}(X_{tk}) = \begin{cases} |X_{tk}|\,, & \text{under } H_1^{tk} \\ G_{\min}|Y_{tk}|\,, & \text{under } H_0^{tk}\,. \end{cases} \tag{44.16}$$

The MMSE-LSA estimator is obtained by using the functions

$$g(\hat{X}_{tk}) = \log|\hat{X}_{tk}|\,,$$
$$\tilde{g}(X_{tk}) = \begin{cases} \log|X_{tk}|\,, & \text{under } H_1^{tk} \\ \log(G_{\min}|Y_{tk}|)\,, & \text{under } H_0^{tk}\,. \end{cases} \tag{44.17}$$

The last two estimators are insensitive to the estimation error of φ_{tk}, the phase of X_{tk}. Therefore, they are combined with the following constrained optimization problem [44.12]:

$$\min_{\hat{\varphi}_{tk}} E\{|\mathrm{e}^{\mathrm{i}\varphi_{tk}} - \mathrm{e}^{\mathrm{i}\hat{\varphi}_{tk}}|^2\} \quad \text{subject to } |\mathrm{e}^{\mathrm{i}\hat{\varphi}_{tk}}| = 1\,. \tag{44.18}$$

This yields an estimator for the complex exponential of the phase, constrained not to affect the spectral magnitude estimate. Alternatively, an estimate for the spectral phase $\hat{\varphi}_{tk}$ can be obtained by minimizing the expected value of the following distortion measure [44.12]

$$d_{\varphi}(\varphi_{tk}, \hat{\varphi}_{tk}) \triangleq 1 - \cos(\varphi_{tk} - \hat{\varphi}_{tk})\,. \tag{44.19}$$

This measure is invariant under modulo 2π transformation of the estimation error $\varphi_{tk} - \hat{\varphi}_{tk}$, and for small estimation errors it closely resembles the squared-error distortion measure, since $1 - \cos\beta \approx \beta^2/2$ for $\beta \ll 1$. The constrained optimization problem (44.18) and the distortion measure (44.19) both yield an estimator $\mathrm{e}^{\mathrm{i}\hat{\varphi}_{tk}} = Y_{tk}/|Y_{tk}|$, which is simply the complex exponential of the noisy signal [44.12].

44.4.1 MMSE Spectral Estimation

Let

$$\xi_{tk} \triangleq \frac{\lambda_{tk}}{\sigma_{tk}^2}\,, \quad \gamma_{\rho tk} \triangleq \frac{Y_{\rho tk}^2}{\sigma_{tk}^2}\,, \tag{44.20}$$

represent the a priori and a posteriori SNRs, respectively ($\rho \in \{\mathrm{R}, \mathrm{I}\}$), and let $G_{\mathrm{MSE}}(\xi, \gamma\rho)$ denote a gain function that satisfies

$$E\left\{X_{\rho tk}\middle|H_1^{tk}, \lambda_{tk}, \sigma_{tk}^2, Y_{\rho tk}\right\} = G_{\mathrm{MSE}}(\xi_{tk}, \gamma_{\rho tk}) Y_{\rho tk}\,. \tag{44.21}$$

Then, substituting (44.15) and (44.21) into (44.5), we have

$$\hat{X}_{tk} = \hat{p}_{tk}\big[G_{\mathrm{MSE}}(\hat{\xi}_{tk}, \hat{\gamma}_{\mathrm{R}tk}) Y_{\mathrm{R}tk} + \mathrm{i} G_{\mathrm{MSE}}(\hat{\xi}_{tk}, \hat{\gamma}_{\mathrm{I}tk}) Y_{\mathrm{I}tk}\big] + (1 - \hat{p}_{tk}) G_{\min} Y_{tk}\,. \tag{44.22}$$

The specific expression for $G_{\mathrm{MSE}}(\xi, \gamma_\rho)$ depends on the particular statistical model:

$$G_{\mathrm{MSE}}(\xi, \gamma_\rho) = \frac{1}{Y_\rho} \int X_\rho\, p(X_\rho|H_1, \lambda, \sigma^2, Y_\rho)\, \mathrm{d}X_\rho = \frac{1}{Y_\rho} \int X_\rho \frac{p(Y_\rho|X_\rho, \sigma^2)\, p(X_\rho|\lambda, H_1)}{p(Y_\rho|\lambda, \sigma^2)}\, \mathrm{d}X_\rho\,.$$

For a Gaussian model, the gain function is independent of the a posteriori SNR. It is often referred to as Wiener

filter, given by [44.5]

$$G_{\mathrm{MSE}}(\xi) = \frac{\xi}{1+\xi} \,. \tag{44.23}$$

For a gamma model, the gain function is given by [44.40]

$$\begin{aligned} G_{\mathrm{MSE}}(\xi, \gamma_\rho) = \frac{1}{\sqrt{8\gamma_\rho}} \Bigg[& \exp\left(\frac{C_{\rho-}^2}{4}\right) D_{-1.5}(C_{\rho-}) \\ & - \exp\left(\frac{C_{\rho+}^2}{4}\right) D_{-1.5}(C_{\rho+}) \Bigg] \\ & \times \Bigg[\exp\left(\frac{C_{\rho-}^2}{4}\right) D_{-0.5}(C_{\rho-}) \\ & + \exp\left(\frac{C_{\rho+}^2}{4}\right) D_{-0.5}(C_{\rho+}) \Bigg]^{-1} , \end{aligned} \tag{44.24}$$

where $C_{\rho+}$ and $C_{\rho-}$ are defined by

$$C_{\rho\pm} \triangleq \frac{\sqrt{3}}{2\sqrt{\xi}} \pm \sqrt{2\gamma_\rho} \,, \tag{44.25}$$

and $D_p(z)$ denotes the parabolic cylinder function [44.44, (9.240)]. For a Laplacian speech model, the gain function is given by [44.45]

$$\begin{aligned} & G_{\mathrm{MSE}}(\xi, \gamma_\rho) \\ & = \frac{1}{\sqrt{\gamma_\rho}} [L_{\rho+}\mathrm{erfcx}(L_{\rho+}) - L_{\rho-}\mathrm{erfcx}(L_{\rho-})] \\ & \quad \times [\mathrm{erfcx}(L_{\rho+}) + \mathrm{erfcx}(L_{\rho-})]^{-1} \,, \end{aligned} \tag{44.26}$$

where $L_{\rho+}$ and $L_{\rho-}$ are defined by

$$L_{\rho\pm} \triangleq \frac{1}{\sqrt{\xi}} \pm \sqrt{\gamma_\rho} \,, \tag{44.27}$$

and $\mathrm{erfcx}(x)$ is the scaled complementary error function, defined by

$$\mathrm{erfcx}(x) \triangleq \mathrm{e}^{x^2} \frac{2}{\sqrt{\pi}} \int_x^\infty \mathrm{e}^{-t^2} \, \mathrm{d}t \,. \tag{44.28}$$

Note that when the signal is surely absent (i. e., when $\hat{p}_{tk} = 0$), the resulting estimator $\hat{X}_{tk}$ reduces to a constant attenuation of Y_{tk} (i. e., $\hat{X}_{tk} = G_{\min} Y_{tk}$). This retains the noise naturalness, and is closely related to the *spectral floor* proposed by *Berouti* et al. [44.6].

Figure 44.5 displays parametric gain curves describing $G_{\mathrm{MSE}}(\xi, \gamma_\rho)$ for several values of γ_ρ, which result from (44.23), (44.24), and (44.26). It shows that the spectral gains are monotonically increasing functions of the a priori SNR when the a posteriori SNR is kept constant. For gamma and Laplacian models, the spectral gains are also monotonically increasing functions of the a posteriori SNR, when the a priori SNR is kept constant.

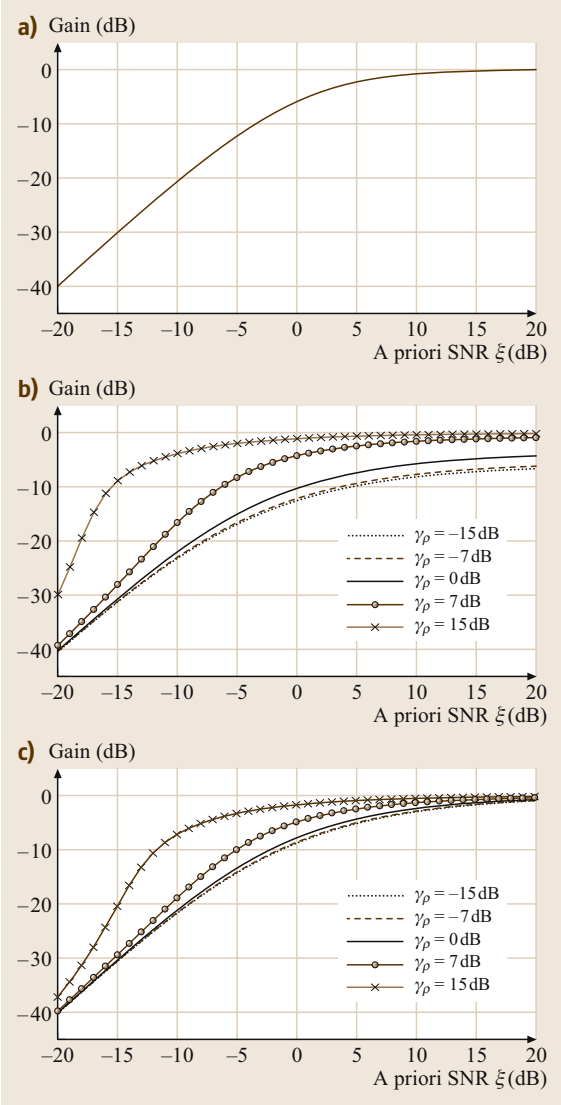


Fig. 44.5a–c Parametric gain curves describing the MMSE gain function $G_{\mathrm{MSE}}(\xi, \gamma_\rho)$ for different speech models. **(a)** Gain curve for a Gaussian model, obtained by (44.23). **(b)** Gain curves for a gamma model, obtained by (44.24). **(c)** Gain curves for a Laplacian model, obtained by (44.26) (after [44.43])

44.4.2 MMSE Log-Spectral Amplitude Estimation

In speech enhancement applications, estimators that minimize the mean-squared error of the log-spectral amplitude have been found advantageous to MMSE spectral estimators [44.12, 17, 46]. An MMSE-LSA estimator is obtained by substituting (44.17) into (44.5). It is difficult, or even impossible, to find analytical expressions for an MMSE-LSA estimator under a gamma or Laplacian model. However, assuming a Gaussian model and combing the resulting amplitude estimate with the phase of the noisy spectral coefficient Y_{tk} yields [44.42]

$$\hat{X}_{tk} = \left[G_{\mathrm{LSA}}(\hat{\xi}_{tk}, \hat{\gamma}_{tk})\right]^{\hat{p}_{tk}} G_{\min}^{1-\hat{p}_{tk}} Y_{tk} \,, \tag{44.29}$$

where $\hat{\gamma}_{tk}$ denotes an estimate for the a posteriori SNR

$$\hat{\gamma}_{tk} = \hat{\gamma}_{\mathrm{R}tk} + \hat{\gamma}_{\mathrm{I}tk} \,, \tag{44.30}$$

and $G_{\mathrm{LSA}}(\xi, \gamma)$ represents the LSA gain function under H_1^{tk} which was derived by *Ephraim* and *Malah* [44.17]

$$G_{\mathrm{LSA}}(\xi, \gamma) \triangleq \frac{\xi}{1+\xi} \exp\left(\frac{1}{2}\int_{\vartheta}^{\infty} \frac{\mathrm{e}^{-x}}{x}\,\mathrm{d}x\right) \,, \tag{44.31}$$

where ϑ is defined by $\vartheta \triangleq \xi\gamma/(1+\xi)$. Similar to the MMSE spectral estimator, the MMSE-LSA estimator reduces to a constant attenuation of Y_{tk} when the signal is surely absent (i. e., $\hat{p}_{tk} = 0$ implies $\hat{X}_{tk} = G_{\min} Y_{tk}$). However, the characteristics of these estimators when the signal is present are readily distinctive. Figure 44.6 displays parametric gain curves describing $G_{\mathrm{LSA}}(\xi, \gamma)$ for several values of γ. For a fixed value of the a posteriori SNR, the LSA gain is a monotonically increasing function of ξ. However, for a fixed value of ξ, the LSA gain is a monotonically *decreasing* function of γ. Note that the gain function $G_{\mathrm{MSE}}(\xi, \gamma_\rho)$ for a Gaussian model is independent of the a posteriori SNR, while for gamma and Laplacian speech models $G_{\mathrm{MSE}}(\xi, \gamma_\rho)$ is an *increasing* function of the a posteriori SNR (Fig. 44.5). The behavior of $G_{\mathrm{LSA}}(\xi, \gamma)$ is related to the useful mechanism that counters the musical noise phenomenon [44.47]. Local bursts of the a posteriori SNR, during noise-only frames, are *pulled down* to the average noise level, thus avoiding local buildup of noise whenever it exceeds its average characteristics. As a result, the MMSE-LSA estimator generally produces lower levels of residual musical noise, when compared with the MMSE spectral estimators.

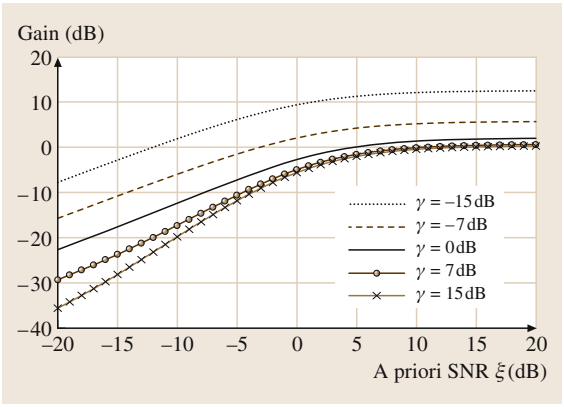


Fig. 44.6 Parametric gain curves describing the MMSE log-spectral amplitude gain function $G_{\mathrm{LSA}}(\xi, \gamma)$ for a Gaussian model, obtained by (44.31)

44.5 Signal Presence Probability Estimation

In this section, we derive an efficient estimator $\hat{p}_{tk|t-1}$ for the a priori speech presence probability. This estimator employs a soft-decision approach to compute three parameters based on the time–frequency distribution of the estimated a priori SNR $\hat{\xi}_{tk}$. The parameters exploit the strong correlation of speech presence in neighboring frequency bins of consecutive frames.

Let ζ_{tk} denote a recursive average of the a priori SNR with a time constant α_ζ,

$$\zeta_{tk} = \alpha_\zeta \zeta_{t-1,k} + (1-\alpha_\zeta)\hat{\xi}_{t-1,k} \,. \tag{44.32}$$

By applying *local* and *global* averaging windows in the frequency domain, we obtain respectively local and global averages of the a priori SNR

$$\zeta_{tk}^{\chi} = \sum_{i=-w_\chi}^{w_\chi} h_\chi(i)\zeta_{t,k-i} \,, \tag{44.33}$$

where the superscript χ designates either *local* or *global*, and h_χ is a normalized window of size $2w_\chi + 1$. We define two parameters, P_{tk}^{local} and P_{tk}^{global}, which represent the relation between the above averages and the like-

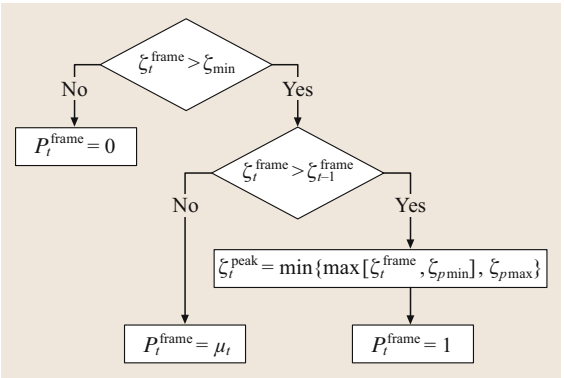


Fig. 44.7 Block diagram for computing P^{frame} (a parameter representing the likelihood of speech in a given frame) (after [44.42])

lihood of speech in the k-th frequency bin of the t-th frame. These parameters are given by

$$P_{tk}^{\chi} = \begin{cases} 0\,, & \text{if } \zeta_{tk}^{\chi} \le \zeta_{\min} \\ 1\,, & \text{if } \zeta_{tk}^{\chi} \ge \zeta_{\max} \\ \frac{\log(\zeta_{tk}^{\chi}/\zeta_{\min})}{\log(\zeta_{\max}/\zeta_{\min})}\,, & \text{otherwise}\,, \end{cases} \tag{44.34}$$

where $\zeta_{\min}$ and $\zeta_{\max}$ are empirical constants, maximized to attenuate noise while maintaining weak speech components.

In order to attenuate noise further in noise-only frames, we define a third parameter, P_t^{frame}, which is based on the speech energy in neighboring frames. An averaging of ζ_{tk} in the frequency domain (possibly over a certain frequency band) yields

$$\zeta_t^{\text{frame}} = \underset{1\le k\le N/2}{\text{mean}}\{\zeta_{tk}\}\,. \tag{44.35}$$

To prevent clipping of speech onsets or weak components, speech is assumed whenever ζ_t^{frame} increases over time. Moreover, the transition from H_1 to H_0 is delayed,

Table 44.1 Values of the parameters used in the implementation of the speech presence probability estimator, for a sampling rate of 16 kHz

$\alpha_\zeta = 0.7$	$\zeta_{\min} = -10\,\text{dB}$	$\zeta_{p\,\min} = 0\,\text{dB}$
$w_{\text{local}} = 1$	$\zeta_{\max} = -5\,\text{dB}$	$\zeta_{p\,\max} = 10\,\text{dB}$
$w_{\text{global}} = 15$	$p_{\min} = 0.005$	
h_{local}, h_{global}: Hann windows		

which reduces the misdetection of weak speech tails, by allowing for a certain decrease in the value of ζ_t^{frame}. Figure 44.7 describes a block diagram for computing P_t^{frame}, where

$$\mu_t \triangleq \begin{cases} 0\,, & \text{if } \zeta_t^{\text{frame}} \le \zeta_t^{\text{peak}}\zeta_{\min} \\ 1\,, & \text{if } \zeta_t^{\text{frame}} \ge \zeta_t^{\text{peak}}\zeta_{\max} \\ \frac{\log(\zeta_t^{\text{frame}}/\zeta_t^{\text{peak}}/\zeta_{\min})}{\log(\zeta_{\max}/\zeta_{\min})}\,, & \text{otherwise}\,, \end{cases} \tag{44.36}$$

represents a soft transition from *speech* to *noise*, ζ_t^{peak} is a confined peak value of ζ_t^{frame}, and $\zeta_{p\,\min}$ and $\zeta_{p\,\max}$ are empirical constants that determine the delay of the transition. Typical values of parameters used for a sampling rate of 16 kHz are summarized in Table 44.1.

The proposed estimate for the a priori speech presence probability is obtained by

$$\hat{p}_{tk|t-1} = P_{tk}^{\text{local}} P_{tk}^{\text{global}} P_t^{\text{frame}}\,. \tag{44.37}$$

Accordingly, $\hat{p}_{tk|t-1}$ is smaller if either previous frames, or recent neighboring frequency bins, do not contain speech. When $\hat{p}_{tk|t-1} \to 0$, the conditional speech presence probability $\hat{p}_{tk} \to 0$ by (44.6), and consequently the signal estimator $\hat{X}_{tk}$ reduces to $\hat{X}_{tk} = G_{\min} Y_{tk}$. Therefore, to reduce the possibility of speech distortion we generally restrict $\hat{p}_{tk|t-1}$ to be larger than a threshold $p_{\min}$ ($p_{\min} > 0$).

44.6 A Priori SNR Estimation

In this section, we address the problem of estimating the speech spectral variance λ_{tk} assuming that $\hat{p}_{tk|t-1}$ and $\widehat{\sigma_{tk}^2}$ are given. We present the decision-directed, causal and noncausal estimators for the a priori SNR $\xi_{tk} = \lambda_{tk}/\sigma_{tk}^2$ under speech presence uncertainty. The a priori SNR ξ_{tk} is estimated for each spectral component and each analysis frame due to the nonstationarity of the speech signal.

44.6.1 Decision-Directed Estimation

Ephraim and *Malah* [44.12] proposed a decision-directed approach, which provides a very useful

estimation method for the a priori SNR [44.47,48]. Accordingly, if speech presence is assumed ($p_{tk} \equiv 1$), then the expression

$$\Xi_{tk} = \alpha \frac{|\hat{X}_{t-1,k}|^2}{\hat{\sigma}^2_{t-1,k}} + (1-\alpha) \max\left\{\hat{\gamma}_{tk} - 1, 0\right\} \tag{44.38}$$

can be substituted for the a priori SNR. The first term, $|\hat{X}_{t-1,k}|^2/\hat{\sigma}^2_{t-1,k}$, represents the a priori SNR resulting from the processing of the previous frame. The second term, $\max\{\hat{\gamma}_{tk} - 1, 0\}$, is a maximum-likelihood estimate for the a priori SNR, based entirely on the current frame. The parameter α ($0 < \alpha < 1$) is a weighting factor that controls the trade-off between noise reduction and transient distortion brought into the signal [44.12, 47]. A larger value of α results in a greater reduction of the musical noise phenomena, but at the expense of attenuated speech onsets and audible modifications of transient components. As a compromise, a value 0.98 of α was determined by simulations and informal listening tests [44.12].

Under speech presence uncertainty, according to [44.12, 49], the expression in (44.38) estimates a *nonconditional a priori* SNR $\eta_{tk} \triangleq E\{|X_{tk}|^2\}/\sigma^2_{tk}$. The a priori SNR $\xi_{tk} = E\{|X_{tk}|^2|H_1^{tk}\}/\sigma^2_{tk}$ is related to η_{tk} by

$$\eta_{tk} = \frac{E\{|X_{tk}|^2|H_1^{tk}\}P(H_1^{tk})}{\sigma^2_{tk}} = \xi_{tk}\, p_{tk|t-1}\,. \tag{44.39}$$

Therefore the estimate for ξ_{tk} should supposedly be given by

$$\hat{\xi}_{tk} = \frac{\Xi_{tk}}{\hat{p}_{tk|t-1}}\,. \tag{44.40}$$

However, the division by $\hat{p}_{tk|t-1}$ may deteriorate the performance of the speech enhancement system [44.50,51]. In some cases, it introduces interaction between the estimated $p_{tk|t-1}$ and the a priori SNR, that adversely affects the total gain for noise-only bins, resulting in an unnaturally structured residual noise [44.52]. To some extent, the noise structuring can be eliminated by utilizing a voice activity detector (VAD) and applying a uniform attenuation factor to frames that do not contain speech [44.49]. Yet, VADs are difficult to tune and their reliability is often insufficient for weak speech components and low input SNR.

Let $\hat{X}_{tk|H_1} = \hat{X}_{tk}|_{\hat{p}_{tk}=1}$ denote an estimate for $\hat{X}_{tk}$ under the hypothesis of speech presence. Then an alternative a priori SNR estimator under speech presence uncertainty is given by [44.42]

$$\hat{\xi}_{tk} = \alpha \frac{|\hat{X}_{t-1,k|H_1}|^2}{\hat{\sigma}^2_{t-1,k}} + (1-\alpha) \max\left\{\hat{\gamma}_{tk} - 1, 0\right\}. \tag{44.41}$$

Notice that for $\hat{p}_{t-1,k|t-2} \neq 1$, this yields a different estimate than either Ξ_{tk} or $\Xi_{tk}/\hat{p}_{tk|t-1}$. In [44.50, 51], it was suggested to simply estimate the a priori SNR by Ξ_{tk}, rather than $\Xi_{tk}/\hat{p}_{tk|t-1}$. However, the use of $\hat{X}_{t-1,k|H_1}$ in (44.41) boosts the gain up when speech is present, which provides a compensation for not dividing by $\hat{p}_{tk|t-1}$.

To show that under speech presence uncertainty it is advantageous to estimate the a priori SNR by the expression in (44.41) rather than by $\Xi_{tk}/\hat{p}_{tk|t-1}$, we assume that an estimate $\hat{p}_{tk|t-1}$ for the a priori speech presence probability is given, and that Ξ_{tk} and $\hat{\xi}_{tk}$ have been calculated by (44.38) and (44.41), respectively. By definition, if H_1^{tk} is true, then the spectral estimate $\hat{X}_{tk}$ should degenerate to $\hat{X}_{tk|H_1}$, and the a priori SNR estimate should coincide with Ξ_{tk}. On the contrary, if H_0^{tk} is true, then $\hat{X}_{tk}$ should reduce to $G_{\min} Y_{tk}$, or equivalently the a priori SNR estimate should be as small as possible. Indeed, if H_1^{tk} is true then

$$\hat{\xi}_{tk}|_{H_1} \approx \Xi_{tk}|_{H_1} \leq \left.\frac{\Xi_{tk}}{\hat{p}_{tk|t-1}}\right|_{H_1}, \tag{44.42}$$

where we have used that under H_1^{tk} the spectral estimate $\hat{X}_{t-1,k}$ is approximately the same as $\hat{X}_{t-1,k|H_1}$ (if H_1^{tk} is true then $H_1^{t-1,k}$ is likely to be true as well, due to the strong correlation of speech presence in successive frames). On the other hand, if H_0^{tk} is true, then $\hat{p}_{tk|t-1}$ is expected to approach zero, and $\hat{\xi}_{tk}$ is likely to be much smaller than $\Xi_{tk}/\hat{p}_{tk|t-1}$:

$$\hat{\xi}_{tk}|_{H_0} \approx \alpha G^2_{\min} \ll \left.\frac{\Xi_{tk}}{\hat{p}_{tk|t-1}}\right|_{H_0} \approx \frac{\alpha G^2_{\min}}{\hat{p}_{tk|t-1}}\,. \tag{44.43}$$

Therefore, under speech presence uncertainty the decision-directed a priori SNR estimator is more favorably modified as in (44.41), rather than dividing Ξ_{tk} by the a priori speech presence probability $\hat{p}_{tk|t-1}$.

44.6.2 Causal Recursive Estimation

In this section, we present a causal conditional estimator $\hat{\xi}_{tk|t} = \hat{\lambda}_{tk|t}/\widehat{\sigma^2_{tk}}$ for the a priori SNR given the noisy measurements up to frame t. The estimator combines two steps, a *propagation* step and an *update* step, following the rational of Kalman filtering, to predict and update the estimate for λ_{tk} recursively as new data arrive.

Let $\mathcal{X}_0^\tau = \{X_{tk} | t = 0, \ldots, \tau, k = 0, \ldots, N-1\}$ represent the set of clean-speech spectral coefficients up to frame τ, and let $\lambda_{tk|\tau} \triangleq E\{|X_{tk}|^2 | H_1^{tk}, \mathcal{X}_0^\tau\}$ denote the *conditional* variance of X_{tk} under H_1^{tk} given the clean spectral coefficients up to frame τ. Assuming that an estimate $\hat{\lambda}_{tk|t-1}$ for the one-frame-ahead conditional variance of X_{tk} is available, an estimate for $\lambda_{tk|t}$ can be obtained by calculating its conditional mean under H_1^{tk} given Y_{tk} and $\hat{\lambda}_{tk|t-1}$. By definition, $\lambda_{tk|t} = |X_{tk}|^2$. Hence,

$$\begin{aligned}\hat{\lambda}_{tk|t} &= E\left\{|X_{tk}|^2 \middle| H_1^{tk}, \hat{\lambda}_{tk|t-1}, Y_{tk}\right\} \\ &= E\left\{X_{\mathrm{R}tk}^2 \middle| H_1^{tk}, \hat{\lambda}_{tk|t-1}, Y_{\mathrm{R}tk}\right\} \\ &\quad + E\left\{X_{\mathrm{I}tk}^2 \middle| H_1^{tk}, \hat{\lambda}_{tk|t-1}, Y_{\mathrm{I}tk}\right\}, \end{aligned} \tag{44.44}$$

where we have used that $X_{\mathrm{R}tk}$ is independent of $Y_{\mathrm{I}tk}$, and $X_{\mathrm{I}tk}$ is independent of $Y_{\mathrm{R}tk}$. Let $G_{\mathrm{SP}}(\xi, \gamma_\rho)$ represent the MMSE gain function in the spectral power domain for $Y_{\rho tk} \neq 0$ [44.43]:

$$E\left\{X_{\rho tk}^2 \middle| H_1^{tk}, \hat{\lambda}_{tk|t-1}, Y_{\rho tk}\right\} = G_{\mathrm{SP}}\left(\hat{\xi}_{tk|t-1}, \gamma_{\rho tk}\right) Y_{\rho tk}^2 , \tag{44.45}$$

where $\rho \in \{\mathrm{R}, \mathrm{I}\}$. Then the specific expression for $G_{\mathrm{SP}}(\xi, \gamma_\rho)$ depends on the particular statistical model. For a Gaussian model, the spectral power gain function is given by

$$G_{\mathrm{SP}}(\xi, \gamma_\rho) = \frac{\xi}{1+\xi}\left(\frac{1}{2\gamma_\rho} + \frac{\xi}{1+\xi}\right) . \tag{44.46}$$

For a gamma model [44.43],

$$\begin{aligned} G_{\mathrm{SP}}(\xi, \gamma_\rho) = \frac{3}{8\gamma_\rho} &\left[\exp\left(\frac{C_{\rho-}^2}{4}\right) D_{-2.5}(C_{\rho-}) \right. \\ &\left. + \exp\left(\frac{C_{\rho+}^2}{4}\right) D_{-2.5}(C_{\rho+})\right] \\ &\times \left[\exp\left(\frac{C_{\rho-}^2}{4}\right) D_{-0.5}(C_{\rho-}) \right. \\ &\left. + \exp\left(\frac{C_{\rho+}^2}{4}\right) D_{-0.5}(C_{\rho+})\right]^{-1} , \end{aligned} \tag{44.47}$$

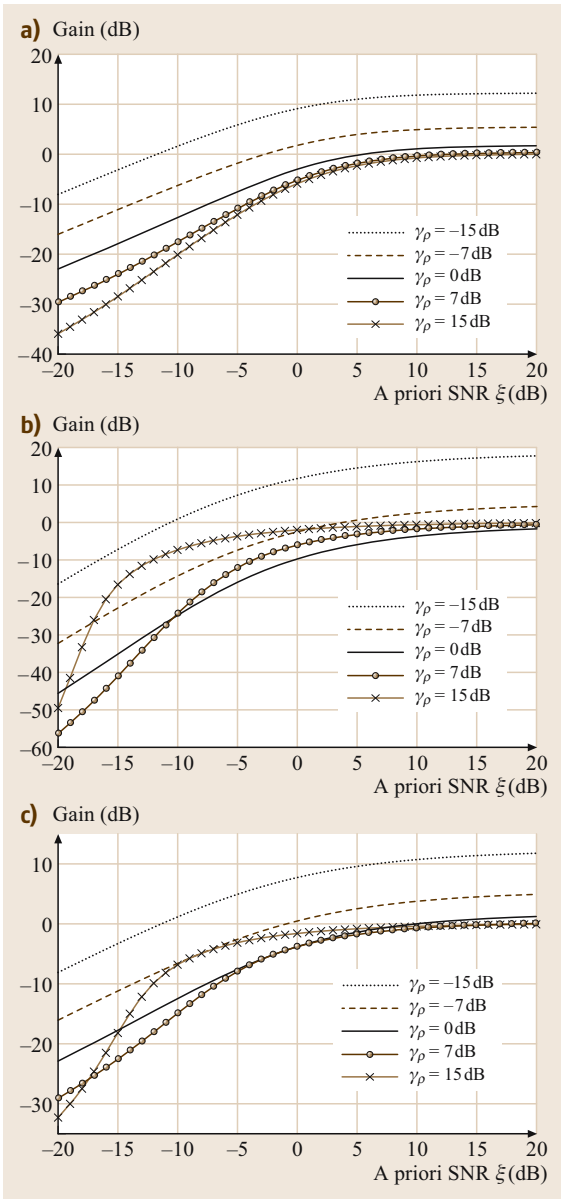


Fig. 44.8a–c Parametric gain curves describing the MMSE spectral power gain function $G_{\mathrm{SP}}(\xi, \gamma_\rho)$ for different speech models. **(a)** Gain curves for a Gaussian model, obtained by (44.46). **(b)** Gain curves for a gamma model, obtained by (44.47). **(c)** Gain curves for a Laplacian model, obtained by (44.48) (after [44.43])

where $C_{\rho\pm}$ are defined by (44.25). For a Laplacian model [44.43],

$$\begin{aligned} &G_{\mathrm{SP}}(\xi,\gamma_\rho) \\ &= \frac{1}{\gamma_\rho}[\mathrm{erfcx}(L_{\rho+})+\mathrm{erfcx}(L_{\rho-})]^{-1} \\ &\quad\times\Bigg[\left(L_{\rho+}^2+0.5\right)\mathrm{erfcx}(L_{\rho+}) \\ &\quad+\left(L_{\rho-}^2+0.5\right)\mathrm{erfcx}(L_{\rho-}) \\ &\quad-\frac{(L_{\rho+}+L_{\rho-})}{\sqrt{\pi}}\Bigg], \end{aligned} \tag{44.48}$$

where $L_{\rho\pm}$ are defined by (44.27). Figure 44.8 shows parametric gain curves, which describe the functions $G_{\mathrm{SP}}(\xi,\gamma_\rho)$ for several values of γ_ρ, resulting from (44.46), (44.47), and (44.48). When γ_ρ is kept constant, $G_{\mathrm{SP}}(\xi,\gamma_\rho)$ is a monotonically increasing function of ξ. When ξ is kept constant, $G_{\mathrm{SP}}(\xi,\gamma_\rho)$ is a monotonically decreasing function of γ_ρ for a Gaussian model, but is not a monotonic function of γ_ρ for gamma or Laplacian models.

Equation (44.45) does not hold in the case $Y_{\rho tk}\to 0$, since $G_{\mathrm{SP}}(\xi,\gamma_\rho)\to\infty$ as $\gamma_\rho\to 0$, and the conditional variance of $X_{\rho tk}$ is generally not zero. For $Y_{\rho tk}=0$ (or practically for $Y_{\rho tk}$ smaller in magnitude than a predetermined threshold) we use the following expressions [44.43]: for a Gaussian model

$$E\left\{X_{\rho tk}^2\middle|H_1^{tk},\hat{\lambda}_{tk|t-1},Y_{\rho tk}=0\right\}=\frac{\hat{\xi}_{tk|t-1}}{1+\hat{\xi}_{tk|t-1}}\sigma_{tk}^2\,, \tag{44.49}$$

for a gamma model

$$\begin{aligned} &E\left\{X_{\rho tk}^2\middle|H_1^{tk},\hat{\lambda}_{tk|t-1},Y_{\rho tk}=0\right\} \\ &=\frac{3D_{-2.5}\left(\frac{\sqrt{3}}{2\sqrt{\hat{\xi}_{tk|t-1}}}\right)}{8D_{-0.5}\left(\frac{\sqrt{3}}{2\sqrt{\hat{\xi}_{tk|t-1}}}\right)}\sigma_{tk}^2 \end{aligned} \tag{44.50}$$

and for a Laplacian model

$$\begin{aligned} &E\left\{X_{\rho tk}^2\middle|H_1^{tk},\hat{\lambda}_{tk|t-1},Y_{\rho tk}=0\right\} \\ &=\sqrt{\frac{2}{\pi}}\frac{\exp\left(\frac{1}{2\hat{\xi}_{tk|t-1}}\right)D_{-3}\left(\sqrt{\frac{2}{\hat{\xi}_{tk|t-1}}}\right)}{\mathrm{erfcx}\left(\frac{1}{\sqrt{\hat{\xi}_{tk|t-1}}}\right)}\sigma_{tk}^2\,. \end{aligned} \tag{44.51}$$

From (44.45–44.51), we can define a function $f(\xi,\sigma^2,Y_\rho^2)$ such that

$$\begin{aligned} &\frac{1}{\sigma_{tk}^2}E\left\{X_{\rho tk}^2\middle|H_1^{tk},\hat{\lambda}_{tk|t-1},Y_{\rho tk}\right\} \\ &=f\left(\hat{\xi}_{tk|t-1},\sigma_{tk}^2,Y_{\rho tk}^2\right) \end{aligned} \tag{44.52}$$

for all $Y_{\rho tk}$. Substituting (44.52) into (44.44), we obtain an estimate for $\xi_{tk|t}$ given by

$$\hat{\xi}_{tk|t}=f\left(\hat{\xi}_{tk|t-1},\widehat{\sigma_{tk}^2},Y_{\mathrm{R}tk}^2\right)+f\left(\hat{\xi}_{tk|t-1},\widehat{\sigma_{tk}^2},Y_{\mathrm{I}tk}^2\right)\,. \tag{44.53}$$

Equation (44.53) is the update step of the recursive estimation, since we start with an estimate $\hat{\xi}_{tk|t-1}$ that relies on the noisy observations up to frame $t-1$, and then update the estimate by using the additional information Y_{tk}.

To formulate the propagation step, we assume that we are given at frame $t-1$ estimates for the speech spectral coefficient $X_{t-1,k}$ and its spectral variance $\lambda_{t-1,k}$, conditioned on $\mathcal{Y}_0^{t-1}=\{Y_{\tau k}|\tau=0,\ldots,t-1,k=0,\ldots,N-1\}$. Then, these estimates can be *propagated* in time to obtain an estimate for λ_{tk}. Since λ_{tk} is correlated with both $\lambda_{t-1,k}$ and $|X_{t-1,k}|$, it was proposed in [44.15] to use an estimate of the form

$$\begin{aligned} &\hat{\lambda}_{tk|t-1} \\ &=\max\left\{(1-\mu)\hat{\lambda}_{t-1,k|t-1}+\mu\left|\hat{X}_{t-1,k|H_1}\right|^2,\lambda_{\min}\right\}, \end{aligned} \tag{44.54}$$

where μ $(0\le\mu\le 1)$ is related to the degree of nonstationarity of the random process $\{\lambda_{tk}|t=0,1,\ldots\}$, and $\lambda_{\min}$ is a lower bound on the variance of X_{tk}. In the case of a pseudostationary process, μ is set to a small value, since $\hat{\lambda}_{tk|t-1}\approx\hat{\lambda}_{t-1,k|t-1}$. In the case of a nonstationary process, μ is set to a larger value, since the variances at successive frames are less correlated, and the relative importance of $\hat{\lambda}_{t-1,k|t-1}$ to predict $\hat{\lambda}_{tk|t-1}$ decreases. Dividing both sides of (44.54) by $\hat{\sigma}_{t-1,k}^2$, we obtain the *propagation* step

$$\begin{aligned} &\hat{\xi}_{tk|t-1} \\ &=\max\left\{(1-\mu)\hat{\xi}_{t-1,k|t-1}+\mu\frac{|\hat{X}_{t-1,k|H_1}|^2}{\hat{\sigma}_{t-1,k}^2},\xi_{\min}\right\}, \end{aligned} \tag{44.55}$$

where $\xi_{\min}$ is a lower bound on the a priori SNR.

The steps of the causal recursive a priori SNR estimation are summarized in Table 44.2. The algorithm is initialized at frame $t=-1$ with $\hat{X}_{-1,k|H_1}=0$ and

Table 44.2 Summary of the causal recursive a priori SNR estimation

Initialization:
$\hat{X}_{-1,k\|H_1} = 0,\ \hat{\xi}_{-1,k\|-1} = \xi_{\min}$, for all k.
For all short-time frames $t = 0, 1, \ldots$
For all frequency bins $k = 0, \ldots, N-1$
Propagation step:
Compute $\hat{\xi}_{tk\|t-1}$ using (44.55)
Update step:
Compute $\hat{\xi}_{tk\|t}$ using (44.53)
Spectral estimation:
Compute $\hat{X}_{tk\|H_1} = \hat{X}_{tk}\big\|_{\hat{p}_{tk}=1}$ using, e.g., (44.22) or (44.29)

$\hat{\xi}_{-1,k|-1} = \xi_{\min}$ for all k. Then, for $t = 0, 1, \ldots$, the propagation and update steps are iterated to obtain estimates for the nonstationary a priori SNR. The spectral gain function employed for the computation of $\hat{X}_{tk|H_1}$ is determined by the particular choice of the distortion measure.

44.6.3 Relation Between Causal Recursive Estimation and Decision-Directed Estimation

The causal conditional estimator $\hat{\xi}_{tk|t}$ for the a priori SNR is closely related to the decision-directed estimator of *Ephraim* and *Malah* [44.12]. The decision-directed estimator under speech presence uncertainty is given by (44.41) where α $(0 \le \alpha \le 1)$ is a weighting factor that controls the trade-off between the noise reduction and the transient distortion introduced into the signal.

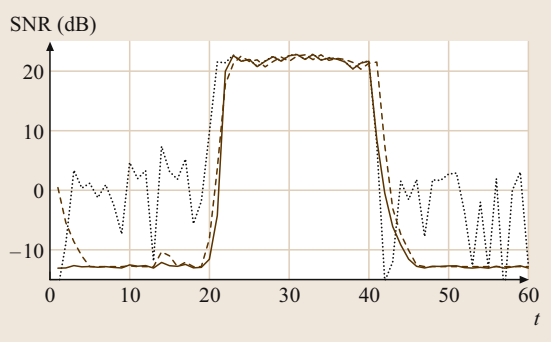


Fig. 44.9 SNRs in successive short-time frames: a posteriori SNR γ_{tk} (*dotted line*), decision-directed a priori SNR $\hat{\xi}^{\mathrm{DD}}_{tk|t}$ (*dashed line*), and causal recursive a priori SNR estimate $\hat{\xi}^{\mathrm{RE}}_{tk|t}$ (*solid line*) (after [44.15])

The update step (44.53) of the causal recursive estimator under a Gaussian model can be written as [44.15]

$$\hat{\xi}_{tk|t} = \alpha_{tk}\hat{\xi}_{tk|t-1} + (1-\alpha_{tk})(\hat{\gamma}_{tk} - 1)\,, \tag{44.56}$$

where α_{tk} is given by

$$\alpha_{tk} \triangleq 1 - \frac{\hat{\xi}^2_{tk|t-1}}{\left(1+\hat{\xi}_{tk|t-1}\right)^2}\,. \tag{44.57}$$

Substituting (44.55) into (44.56) and (44.57) with $\mu \equiv 1$, and applying the lower-bound constraint to $\hat{\xi}_{tk|t}$ rather than $\hat{\xi}_{tk|t-1}$, we have

$$\hat{\xi}_{tk|t} = \max\left\{\alpha_{tk}\frac{\left|\hat{X}_{t-1,k|H_1}\right|^2}{\hat{\sigma}^2_{t-1,k}} + (1-\alpha_{tk})\left(\hat{\gamma}_{tk}-1\right), \xi_{\min}\right\}\,, \tag{44.58}$$

$$\alpha_{tk} = 1 - \frac{\left|\hat{X}_{t-1,k|H_1}\right|^4}{\left(\hat{\sigma}^2_{t-1,k} + \left|\hat{X}_{t-1,k|H_1}\right|^2\right)^2}\,. \tag{44.59}$$

The expression (44.58) with $\alpha_{tk} \equiv \alpha$ is actually a practical form of the decision-directed estimator,

$$\hat{\xi}^{\mathrm{DD}}_{tk|t} = \max\left\{\alpha\frac{\left|\hat{X}_{t-1,k|H_1}\right|^2}{\hat{\sigma}^2_{t-1,k}} + (1-\alpha)\left(\hat{\gamma}_{tk}-1\right), \xi_{\min}\right\}\,, \tag{44.60}$$

that includes a lower-bound constraint to further reduce the level of residual musical noise [44.47]. Accordingly, a special case of the causal recursive estimator with $\mu \equiv 1$ degenerates to a *decision-directed* estimator with a *time-varying frequency-dependent* weighting factor α_{tk}.

It is interesting to note that the weighting factor α_{tk}, given by (44.59), is monotonically decreasing as a function of the instantaneous SNR, $|\hat{X}_{t-1,k|H_1}|^2/\hat{\sigma}^2_{t-1,k}$. A decision-directed estimator with a larger weighting factor is indeed preferable during speech absence (to reduce musical noise phenomena), while a smaller weighting factor is more advantageous during speech presence (to reduce signal distortion) [44.47]. The above special case of the causal recursive estimator conforms to such a desirable behavior. Moreover, the general form of the causal recursive estimator provides an additional degree of freedom for adjusting the value of μ in (44.55) to the degree of spectral nonstationarity. This may produce even further improvement in the performance.

The different behaviors of the causal recursive estimator $\hat{\xi}^{\mathrm{RE}}_{tk|t}$ (Table 44.2) and the decision-directed estimator $\hat{\xi}^{\mathrm{DD}}_{tk|t}$ (44.60) are illustrated in the example

of Fig. 44.9. The analyzed signal contains only white Gaussian noise during the first and last 20 frames, and in between it contains an additional sinusoidal component at the displayed frequency with 0 dB SNR. (Note that the SNR is computed in the time domain, whereas the a priori and a posteriori SNRs are computed in the time–frequency domain. Therefore, the latter SNRs may increase at the displayed frequency well above the average SNR.) The signal is transformed into the STFT domain using half-overlapping Hamming windows. The a priori SNR estimates, $\hat{\xi}^{\mathrm{RE}}_{tk|t}$ and $\hat{\xi}^{\mathrm{DD}}_{tk|t}$, are obtained by using the parameters $\xi_{\min} = -25\,\mathrm{dB}$, $\alpha = 0.98$, $\mu = 0.9$. The spectral estimate $\hat{X}_{tk|H_1}$ is recursively obtained by applying $G_{\mathrm{LSA}}(\hat{\xi}_{tk|t}, \hat{\gamma}_{tk})$ to the noisy spectral measurements (44.29, 31).

Figure 44.9 shows that, when the a posteriori SNR γ_{tk} is sufficiently low, the causal recursive a priori SNR estimate is smoother than the decision-directed estimate, which helps reducing the level of musical noise. When γ_{tk} increases, the response of the a priori SNR $\hat{\xi}^{\mathrm{RE}}_{tk|t}$ is initially slower than $\hat{\xi}^{\mathrm{DD}}_{tk|t}$, but then builds up faster to the a posteriori SNR. When γ_{tk} is sufficiently high, $\hat{\xi}^{\mathrm{DD}}_{tk|t}$ follows the a posteriori SNR with a delay of one frame, whereas $\hat{\xi}^{\mathrm{RE}}_{tk|t}$ follows the a posteriori SNR instantaneously. When γ_{tk} decreases, the response of $\hat{\xi}^{\mathrm{RE}}_{tk|t}$ is immediate, while that of $\hat{\xi}^{\mathrm{DD}}_{tk|t}$ is delayed by one frame. As a consequence, when compared with the decision-directed estimator, the causal recursive estimator produces a lower level of musical noise while not increasing the audible distortion in the enhanced signal [44.15].

44.6.4 Noncausal Recursive Estimation

In some important applications, e.g., digital voice recording, surveillance, speech recognition and speaker identification, a delay of a few short-term frames between the enhanced speech and the noisy observation is tolerable. In such cases, a noncausal estimation approach may produce less signal distortion and less musical residual noise than a causal estimation approach. In this section, we present a noncausal conditional estimator $\hat{\xi}_{tk|t+L}$ for the a priori SNR, given the noisy measurements up to frame $t+L$, where $L > 0$ denotes the admissible time delay in frames. Similar to the causal estimator, the noncausal estimator combines update and propagation steps to recursively estimate λ_{tk} as new data arrive. However, future spectral measurements are also employed in the process to better predict the spectral variances of the clean speech.

Let $\lambda'_{tk|t+L} \triangleq E\{|X_{tk}|^2 | \mathcal{Y}_0^{t-1}, \mathcal{Y}_{t+1}^{t+L}\}$ denote the conditional spectral variance of X_{tk} given $\mathcal{Y}_0^{t+L}$ excluding the noisy measurement at frame t. Let $\lambda_{tk|[t+1,t+L]} \triangleq E\{|X_{tk}|^2 | \mathcal{Y}_{t+1}^{t+L}\}$ denote the conditional spectral variance of X_{tk} given the subsequent noisy measurements $\mathcal{Y}_{t+1}^{t+L}$. Then, similar to (44.53), the estimate for ξ_{tk} given $\hat{\xi}'_{tk|t+L} \triangleq \hat{\lambda}'_{tk|t+L}/\widehat{\sigma_{tk}^2}$ and Y_t can be updated by

$$\hat{\xi}_{tk|t+L} = f\left(\hat{\xi}'_{tk|t+L}, \widehat{\sigma_{tk}^2}, Y^2_{\mathrm{R}tk}\right) + f\left(\hat{\xi}'_{tk|t+L}, \widehat{\sigma_{tk}^2}, Y^2_{\mathrm{I}tk}\right) . \tag{44.61}$$

To obtain an estimate for $\lambda'_{tk|t+L}$, we employ the estimates $\hat{X}_{t-1,k|H_1}$ and $\hat{\lambda}_{t-1,k|t+L-1}$ from the previous frame, and derive an estimate for λ_{tk} from the measurements $\mathcal{Y}_{t+1}^{t+L}$. Suppose an estimate $\hat{\lambda}_{tk|[t+1,t+L]}$ is given, we propagate the estimates from frame $t-1$ to frame t by [44.15, 53]

$$\hat{\lambda}'_{tk|t+L} = \max\left\{\mu\left|\hat{X}_{t-1,k|H_1}\right|^2 + (1-\mu)\left[\mu'\hat{\lambda}_{t-1,k|t+L-1} + (1-\mu')\hat{\lambda}_{tk|[t+1,t+L]}\right], \lambda_{\min}\right\} , \tag{44.62}$$

where μ ($0 \le \mu \le 1$) is related to the stationarity of the random process $\{\lambda_{tk} | t = 0, 1, \ldots\}$, and μ' ($0 \le \mu' \le 1$) is associated with the reliability of the estimate $\hat{\lambda}_{tk|[t+1,t+L]}$ in comparison with that of $\hat{\lambda}_{t-1,k|t+L-1}$. Dividing both sides of (44.62) by $\hat{\sigma}^2_{t-1,k}$, we have the following *backward–forward propagation* step:

$$\hat{\xi}'_{tk|t+L} = \max\left\{\mu\frac{\left|\hat{X}_{t-1,k|H_1}\right|^2}{\hat{\sigma}^2_{t-1,k}} + (1-\mu)\left[\mu'\hat{\xi}_{t-1,k|t+L-1} + (1-\mu')\hat{\xi}_{tk|[t+1,t+L]}\right], \xi_{\min}\right\} . \tag{44.63}$$

An estimate for the a priori SNR ξ_{tk} given the measurements $\mathcal{Y}_{t+1}^{t+L}$ is obtained by

$$\hat{\xi}_{tk|[t+1,t+L]} = \begin{cases} \frac{1}{L}\sum_{n=1}^{L}\hat{\gamma}_{t+n,k} - \beta_f , & \text{if positive} , \\ 0 , & \text{otherwise} , \end{cases} \tag{44.64}$$

where β_f ($\beta_f \ge 1$) is an oversubtraction factor to compensate for a sudden increase in the noise level. This estimator is an anticausal version of the maximum-likelihood a priori SNR estimator suggested in [44.12].

Table 44.3 Summary of the noncausal recursive a priori SNR estimation

Initialization:
$\hat{X}_{-1,k\|H_1} = 0,\ \hat{\xi}_{-1\|L-1} = \xi_{\min}$.
For all short-time frames $t = 0, 1, \ldots$
For all frequency bins $k = 0, \ldots, N-1$
Backward estimation:
Compute $\hat{\xi}_{tk\|[t+1,t+L]}$ using (44.64)
Backward–forward propagation:
Compute $\hat{\xi}'_{tk\|t+L}$ using (44.63)
Update step:
Compute $\hat{\xi}_{tk\|t+L}$ using (44.61)
Spectral estimation:
Compute $\hat{X}_{tk\|H_1} = \hat{X}_{tk}\big\|_{\hat{p}_{tk}=1}$ using, e.g., (44.22) or (44.29)

The steps of the noncausal recursive a priori SNR estimation are summarized in Table 44.3. The algorithm is initialized at frame $t = -1$ with $\hat{X}_{-1,k|H_1} = 0$ and $\hat{\xi}_{-1|L-1} = \xi_{\min}$. Then, for $t = 0, 1, \ldots$, the propagation and update steps are iterated to obtain estimates for the a priori SNR and the speech spectral components.

Figure 44.10 demonstrates the behavior of the noncausal recursive estimator in the same example of Fig. 44.9. The noncausal a priori SNR estimate $\hat{\xi}^{\mathrm{RE}}_{tk|t+3}$ is obtained with the parameters $\xi_{\min} = -25\,\mathrm{dB}$, $\mu = \mu' = 0.9$, $\beta_f = 2$, and $L = 3$ frames delay. A comparison of Figs. 44.9 and 44.10 indicates that the differences between the causal and noncausal recursive estimators are primarily noticeable during onsets

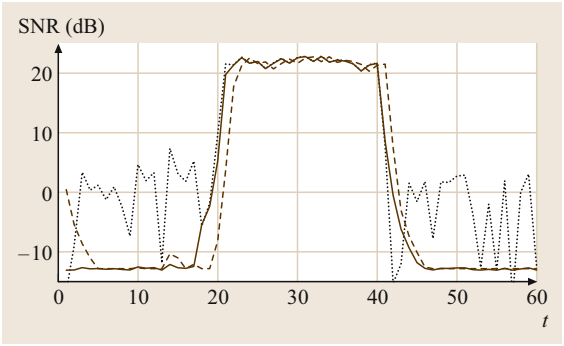


Fig. 44.10 SNRs in successive short-time frames: a posteriori SNR γ_{tk} (*dotted line*), decision-directed a priori SNR $\hat{\xi}^{\mathrm{DD}}_{tk|t}$ (*dashed line*), and noncausal recursive a priori SNR estimate $\hat{\xi}^{\mathrm{RE}}_{tk|t+3}$ with three-frame delay (*solid line*) (after [44.15])

of signal components. Clearly, the *causal* a priori SNR estimator, as well as the decision-directed estimator, cannot respond too fast to an abrupt increase in γ_{tk}, since it necessarily implies an increase in the level of musical residual noise. By contrast, the *noncausal* estimator, having a few subsequent spectral measurements at hand, is capable of discriminating between speech onsets and irregularities in γ_{tk} corresponding to noise only. Therefore, in comparison with the decision-directed estimator, the noncausal a priori SNR estimator produces even lower levels of musical noise and signal distortion [44.15, 53].

44.7 Noise Spectrum Estimation

In this section, we derive an estimator for the noise power spectrum under speech presence uncertainty. The noise estimate is obtained by averaging past spectral power values of the noisy measurement, and multiplying the result by a constant factor that compensates the bias. The recursive averaging is carried out using a time-varying frequency-dependent smoothing parameter that is adjusted by the speech presence probability.

44.7.1 Time-Varying Recursive Averaging

A common noise estimation technique is to average past spectral power values of the noisy measurement recursively during periods of speech absence, and hold the estimate during speech presence. Specifically,

$$H_0^{tk} : \overline{\sigma}^2_{t+1,k} = \alpha_{\mathrm{d}} \overline{\sigma}^2_{tk} + (1-\alpha_{\mathrm{d}})|Y_{tk}|^2$$
$$H_1^{tk} : \overline{\sigma}^2_{t+1,k} = \overline{\sigma}^2_{tk}\,, \qquad (44.65)$$

where α_{d} $(0 < \alpha_{\mathrm{d}} < 1)$ denotes a smoothing parameter. Under speech presence uncertainty, we can employ the conditional speech presence probability, and carry out the recursive averaging by

$$\overline{\sigma}^2_{t+1,k} = \tilde{p}_{tk}\overline{\sigma}^2_{tk} + (1-\tilde{p}_{tk})\left[\alpha_{\mathrm{d}}\overline{\sigma}^2_{tk} + (1-\alpha_{\mathrm{d}})|Y_{tk}|^2\right], \qquad (44.66)$$

where $\tilde{p}_{tk}$ is an estimator for the conditional speech presence probability $p_{tk} = P(H_1^{tk}|Y_{tk})$. Equivalently, the

recursive averaging can be obtained by

$$\overline{\sigma}^2_{t+1,k} = \tilde{\alpha}_{tk}\overline{\sigma}^2_{tk} + \big(1-\tilde{\alpha}_{tk}\big)|Y_{tk}|^2 \,, \tag{44.67}$$

where

$$\tilde{\alpha}_{tk} \triangleq \alpha_{\mathrm{d}} + (1-\alpha_{\mathrm{d}})\tilde{p}_{tk} \tag{44.68}$$

is a time-varying frequency-dependent smoothing parameter. The smoothing parameter $\tilde{\alpha}_{tk}$ is adjusted by the speech presence probability, which is estimated based on the noisy measurement.

Here we make a distinction between the estimator $\hat{p}_{tk}$ in (44.3), used for estimating the clean speech, and the estimator $\tilde{p}_{tk}$, which controls the adaptation of the noise spectrum. Clearly, deciding speech is absent (H_0) when speech is present (H_1) is more destructive when estimating the speech than when estimating the noise. Hence, different decision rules are employed [44.42], and generally we tend to employ estimators that satisfy $\hat{p}_{tk} \geq \tilde{p}_{tk}$. Given an estimator $\tilde{p}_{tk|t-1}$ for the a priori speech presence probability, the conditional speech presence probability estimate can be obtained from Bayes' rule, which under a Gaussian model reduces to [44.12]

$$\tilde{p}_{tk} = \left[1 + \frac{\big(1-\tilde{p}_{tk|t-1}\big)\big(1+\hat{\xi}_{tk}\big)}{\tilde{p}_{tk|t-1}\exp\left(\frac{\hat{\xi}_{tk}\hat{\gamma}_{tk}}{1+\hat{\xi}_{tk}}\right)}\right]^{-1} . \tag{44.69}$$

In the next subsection we present an estimator $\tilde{p}_{tk|t-1}$ that enables noise spectrum estimation during speech activity. Both $\tilde{p}_{tk|t-1}$ and $\hat{p}_{tk|t-1}$ are biased toward higher values, since deciding that speech is absent when speech is present results ultimately in the attenuation of speech components. Whereas, the alternative false decision, up to a certain extent, merely introduces some level of residual noise. Accordingly, we include a bias compensation factor in the noise estimator

$$\hat{\sigma}^2_{t+1,k} = \beta\overline{\sigma}^2_{t+1,k} \tag{44.70}$$

such that the factor β ($\beta \geq 1$) compensates the bias when speech is absent

$$\beta \triangleq \left.\frac{\sigma^2_{tk}}{E\big\{\overline{\sigma}^2_{tk}\big\}}\right|_{\xi_{tk}=0} . \tag{44.71}$$

The value of β is completely determined by the particular estimator for the a priori speech absence probability [44.54]. We note that the noise estimate is based on a variable time segment in each subband, which takes into account the probability of speech presence. The time segment is longer in subbands that contain frequent *speech* portions, and shorter in subbands that contain frequent *silence* portions. This feature has been considered [44.55] a desirable characteristic of the noise estimator, which improves its robustness and tracking capability.

44.7.2 Minima-Controlled Estimation

In this section, we present an estimator $\tilde{p}_{tk|t-1}$ that is controlled by the minima values of a smoothed power spectrum of the noisy signal. In contrast to the minimum statistics (MS) and related methods [44.56, 57], the smoothing of the noisy power spectrum is carried out in both time and frequency. This takes into account the strong correlation of speech presence in neighboring frequency bins of consecutive frames [44.42]. Furthermore, the procedure comprises two iterations of smoothing and minimum tracking. The first iteration provides a rough voice activity detection in each frequency band. Then, the smoothing in the second iteration excludes relatively strong speech components, which makes the minimum tracking during speech activity robust, even when using a relatively large smoothing window. A larger smoothing window decreases the variance of the minima values, but also widens the peaks of the speech activity power. An alternative solution is to modify the smoothing in time and frequency based on a smoothed a posteriori SNR [44.56].

Let α_{s} ($0 < \alpha_{\mathrm{s}} < 1$) be a smoothing parameter, and let b denote a normalized window function of length $2w+1$, i. e., $\sum_{i=-w}^{w} b_i = 1$. The frequency smoothing of the noisy power spectrum in each frame is defined by

$$S^{\mathrm{f}}_{tk} = \sum_{i=-w}^{w} b_i |Y_{t,k-i}|^2 \,. \tag{44.72}$$

Subsequently, smoothing in time is performed by a first-order recursive averaging:

$$S_{tk} = \alpha_{\mathrm{s}} S_{t-1,k} + (1-\alpha_{\mathrm{s}}) S^{\mathrm{f}}_{tk} \,. \tag{44.73}$$

In accordance with the MS method, the minima values of S_{tk} are picked within a finite window of length D, for each frequency bin:

$$S^{\min}_{tk} \triangleq \min\{S_{\tau k} \mid t-D+1 \leq \tau \leq t\} \,. \tag{44.74}$$

It follows [44.56] that there exists a constant factor $B_{\min}$, independent of the noise power spectrum, such that

$$E\big\{S^{\min}_{tk} \big| \xi_{tk} = 0\big\} = B^{-1}_{\min}\sigma^2_{tk} \,. \tag{44.75}$$

The factor $B_{\min}$ represents the bias of a minimum noise estimate, and generally depends on the values of D, α_{s},

w and the spectral analysis parameters (type, length and overlap of the analysis windows). The value of $B_{\min}$ can be estimated by generating a white Gaussian noise, and computing the inverse of the mean of $S_{tk}^{\min}$. This also takes into account the time–frequency correlation of the noisy periodogram $|Y_{tk}|^2$. Notice that the value of $B_{\min}$ is fixed, whereas in [44.56] it is estimated for each frequency band and each frame.

Let $\gamma_{tk}^{\min}$ and ζ_{tk} be defined by

$$\gamma_{tk}^{\min} \triangleq \frac{|Y_{tk}|^2}{B_{\min} S_{tk}^{\min}} ,$$
$$\zeta_{tk} \triangleq \frac{S_{tk}}{B_{\min} S_{tk}^{\min}} . \tag{44.76}$$

Under a Gaussian model, the probability density functions of $\gamma_{tk}^{\min}$ and ζ_{tk}, in the absence of speech, can be approximated by exponential and chi-square densities, respectively [44.54]:

$$p\left(\gamma_{tk}^{\min} \mid H_0^{tk}\right) \approx \mathrm{e}^{-\gamma_{tk}^{\min}} u\left(\gamma_{tk}^{\min}\right) , \tag{44.77}$$

$$p\left(\zeta_{tk} \mid H_0^{tk}\right) \approx \frac{1}{\left(\frac{2}{m}\right)^{m/2} \Gamma\left(\frac{m}{2}\right)} \zeta_{tk}^{m/2-1} \times \exp\left(-\frac{m\zeta_{tk}}{2}\right) u(\zeta_{tk}) , \tag{44.78}$$

where $\Gamma(\cdot)$ is the gamma function, and m is the equivalent degrees of freedom. Based on the first iteration smoothing and minimum tracking, a rough decision about speech presence is given by

$$I_{tk} = \begin{cases} 1 , & \text{if } \gamma_{tk}^{\min} < \gamma_0 \text{ and } \zeta_{tk} < \zeta_0 \\ & \text{(speech is absent)} \\ 0 , & \text{otherwise} \\ & \text{(speech is present)} . \end{cases} \tag{44.79}$$

The thresholds γ_0 and ζ_0 are set to satisfy a certain significance level ϵ:

$$\mathcal{P}\left(\gamma_{tk}^{\min} \geq \gamma_0 \mid H_0^{tk}\right) < \epsilon , \tag{44.80}$$

$$\mathcal{P}\left(\zeta_{tk} \geq \zeta_0 \mid H_0^{tk}\right) < \epsilon . \tag{44.81}$$

From (44.77) and (44.78) we have

$$\gamma_0 = -\log(\epsilon) , \tag{44.82}$$

$$\zeta_0 = \frac{1}{m} F_{\chi^2;m}^{-1}(1-\epsilon) , \tag{44.83}$$

where $F_{\chi^2;m}(x)$ denotes the standard chi-square cumulative distribution function, with m degrees of freedom. Typically, we use $\epsilon = 0.01$ and $m = 32$, so $\gamma_0 = 4.6$ and $\zeta_0 = 1.67$.

The second iteration of smoothing is conditional on the rough speech activity detection of the first iteration. It includes only the power spectral components, which have been identified as containing primarily noise. We set the initial condition for the first frame by $\tilde{S}_{0,k} = S_{0,k}^{\mathrm{f}}$. Then, for $t > 0$ the smoothing in frequency, employing the above voice activity detector, is obtained by

$$\tilde{S}_{tk}^{\mathrm{f}} = \begin{cases} \dfrac{\sum_{i=-w}^{w} b_i I_{t,k-i} |Y_{t,k-i}|^2}{\sum_{i=-w}^{w} b_i I_{t,k-i}} , & \text{if } \sum_{i=-w}^{w} I_{t,k-i} \neq 0 \\ \tilde{S}_{t-1,k} , & \text{otherwise} . \end{cases} \tag{44.84}$$

Smoothing in time is given, as before, by a first-order recursive averaging

$$\tilde{S}_{tk} = \alpha_{\mathrm{s}} \tilde{S}_{t-1,k} + (1-\alpha_{\mathrm{s}}) \tilde{S}_{tk}^{\mathrm{f}} . \tag{44.85}$$

The minima values of $\tilde{S}_{tk}$ are picked within a finite window of length D, for each frequency bin

$$\tilde{S}_{tk}^{\min} \triangleq \min\left\{\tilde{S}_{\tau k} \mid t - D + 1 \leq \tau \leq t\right\} .$$

Accordingly, $\tilde{S}_{tk}^{\min}$ represents minima tracking that is conditional on the rough speech activity detection of the first iteration. We note that keeping the strong speech components out of the smoothing process enables improved minimum tracking. In particular, a larger smoothing parameter (α_{s}) and smaller minima search window (D) can be used. This reduces the variance of the minima values [44.56], and shortens the delay when responding to a rising noise power, which eventually improves the tracking capability of the noise estimator.

Let $\tilde{\gamma}_{tk}^{\min}$ and $\tilde{\zeta}_{tk}$ be defined by

$$\tilde{\gamma}_{tk}^{\min} \triangleq \frac{|Y_{tk}|^2}{B_{\min} \tilde{S}_{tk}^{\min}} ,$$
$$\tilde{\zeta}_{tk} \triangleq \frac{S_{tk}}{B_{\min} \tilde{S}_{tk}^{\min}} . \tag{44.86}$$

Since we use a relatively small significance level in the first iteration ($\epsilon = 0.01$), the influence of the voice activity detector in noise-only periods can be neglected. That is, the effect of excluding strong *noise* components from the smoothing process is negligible. Accordingly, the conditional distributions of $\tilde{\gamma}_{tk}^{\min}$ and $\tilde{\zeta}_{tk}$, in the absence of speech, are approximately the same as those of $\gamma_{tk}^{\min}$ and ζ_{tk} (44.77, 78).

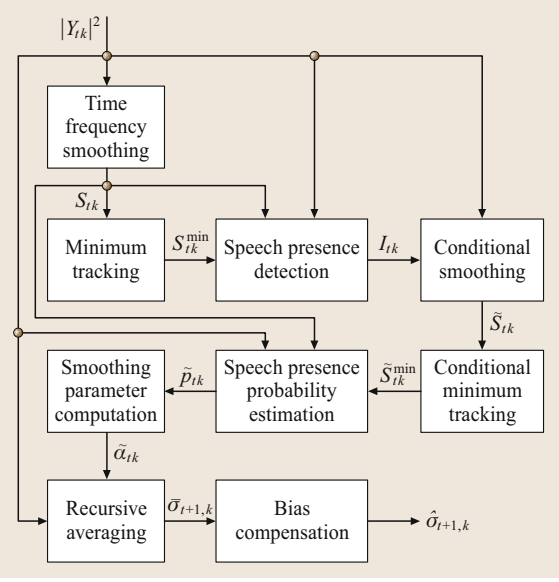


Fig. 44.11 Block diagram of the IMCRA noise spectrum estimation

An estimator for the a priori speech presence probability is given by

$$\tilde{p}_{tk|t-1} = \begin{cases} 0\,, & \text{if } \tilde{\gamma}_{tk}^{\min} \le 1 \\ & \text{and } \tilde{\zeta}_{tk} < \zeta_0 \\ \left(\tilde{\gamma}_{tk}^{\min} - 1\right)/(\gamma_1 - 1)\,, & \text{if } 1 < \tilde{\gamma}_{tk}^{\min} < \gamma_1 \\ & \text{and } \tilde{\zeta}_{tk} < \zeta_0 \\ 1\,, & \text{otherwise}\,. \end{cases} \tag{44.87}$$

The threshold γ_1 is set to satisfy a certain significance level ϵ_1 ($\epsilon_1 > \epsilon$):

$$P\left(\tilde{\gamma}_{tk}^{\min} > \gamma_1 \middle| H_0^{tk}\right) < \epsilon_1 \quad \Rightarrow \quad \gamma_1 \approx -\log(\epsilon_1)\,. \tag{44.88}$$

Typically $\epsilon_1 = 0.05$, and $\gamma_1 = 3$.

The a priori speech presence probability estimator assumes that speech is present ($\tilde{p}_{tk|t-1} = 1$) whenever $\tilde{\zeta}_{tk} \ge \zeta_0$ or $\tilde{\gamma}_{tk}^{\min} \ge \gamma_1$. That is, whenever the local measured power, S_{tk}, or the instantaneous measured power, $|Y_{tk}|^2$, are relatively high compared to the noise power $B_{\min}\tilde{S}_{tk}^{\min} \approx \sigma_{tk}^2$. The estimator assumes that speech is absent ($\tilde{p}_{tk|t-1} = 0$) whenever both the local and instantaneous measured powers are relatively low compared to the noise power ($\tilde{\gamma}_{tk}^{\min} \le 1$ and $\tilde{\zeta}_{tk} < \zeta_0$). In between, the estimator provides a soft transition between speech absence and speech presence, based on the value of $\tilde{\gamma}_{tk}^{\min}$.

The main objective of combining conditions on both $\tilde{\gamma}_{tk}^{\min}$ and $\tilde{\zeta}_{tk}$ is to prevent an increase in the estimated noise during weak speech activity, especially when the input SNR is low. Weak speech components can often be extracted using the condition on $\tilde{\zeta}_{tk}$. Sometimes, speech components are so weak that $\tilde{\zeta}_{tk}$ is smaller than ζ_0. In that case, most of the speech power is still excluded from the averaging process using the condition on $\tilde{\gamma}_{tk}^{\min}$. The remaining speech components can hardly affect the noise estimator, since their power is relatively low compared to that of the noise.

A block diagram of the improved minima-controlled recursive averaging (IMCRA) [44.54] noise spectrum estimation is described in Fig. 44.11. Typical values of parameters used in the implementation of the IMCRA noise estimator for a sampling rate of 16 kHz are summarized in Table 44.4. The noise spectrum estimate, $\hat{\sigma}_{tk}^2$, is initialized at the first frame by $\hat{\sigma}_{0,k}^2 = |Y_{0,k}|^2$. Then, at each frame t ($t \ge 0$), it is used, together with the current observation Y_{tk}, for estimating the noise power spectrum at the next frame, $t+1$. The bias compensation factor β is given by [44.54]

$$\beta = \frac{\gamma_1 - 1 - \mathrm{e}^{-1} + \mathrm{e}^{-\gamma_1}}{\gamma_1 - 1 - 3\,\mathrm{e}^{-1} + (\gamma_1 + 2)\,\mathrm{e}^{-\gamma_1}}\,. \tag{44.89}$$

In particular, for $\gamma_1 = 3$, we have $\beta = 1.47$. The value of $\tilde{\alpha}_{tk}$ is updated for each frequency bin and time frame, using the speech presence probability $\tilde{p}_{tk}$, and (44.68).

Table 44.4 Values of the parameters used in the implementation of the IMCRA noise estimator, for a sampling rate of 16 kHz

$w = 1$	$\alpha_s = 0.9$	$\alpha_d = 0.85$	$\beta = 1.47$
$B_{\min} = 1.66$	$\zeta_0 = 1.67$	$\gamma_0 = 4.6$	$\gamma_1 = 3$
$D = 120$	b: Hann window		

44.8 Summary of a Spectral Enhancement Algorithm

In this section, we present an example of a speech enhancement algorithm, which is based on an MMSE log-spectral amplitude estimation under a Gaussian model, IMCRA noise estimation, and decision-directed

Table 44.5 Summary of a speech enhancement algorithm

Initialization at the first frame for all frequency-bins $k = 1, \ldots, N/2$:
$\hat{\sigma}_{0k}^2 = \lvert Y_{0k}\rvert^2;\ \overline{\sigma}_{0k}^2 = \lvert Y_{0k}\rvert^2;\ S_{0k} = S_{0k}^{\mathrm{f}};\ S_{0k}^{\min} = S_{0k}^{\mathrm{f}};\ S_k^{\min\_\mathrm{sw}} = S_{0k}^{\mathrm{f}};\ \tilde{S}_{0k} = S_{0k}^{\mathrm{f}};\ \tilde{S}_{0k}^{\min} = S_{0k}^{\mathrm{f}};\ \tilde{S}_k^{\min\_\mathrm{sw}} = S_{0k}^{\mathrm{f}};\ \zeta_{0k} = 0.$
Let $\ell = 0$. % ℓ is a counter for frames within a subwindow ($0 \le \ell \le V$).
For all short-time frames $t = 0, 1, \ldots$
For all frequency bins $k = 1, \ldots, N/2$
Compute the a posteriori SNR $\hat{\gamma}_{tk}$ using (44.20) and (44.30), and the a priori SNR $\hat{\xi}_{tk}$ using (44.41), with the initial condition $\hat{\xi}_{0k} = \alpha + (1-\alpha)\max\{\hat{\gamma}_{0k} - 1, 0\}$.
Compute the conditional spectral estimate under the hypothesis of speech presence $\hat{X}_{tk\lvert H_1} = G_{\mathrm{LSA}}(\hat{\xi}_{tk}, \hat{\gamma}_{tk}) Y_{tk}$ using (44.29) and (44.31).
Compute the smoothed power spectrum S_{tk} using (44.72) and (44.73), and update its running minimum: $S_{tk}^{\min} = \min\{S_{t-1,k}^{\min}, S_{tk}\};\ S_k^{\min\_\mathrm{sw}} = \min\{S_k^{\min\_\mathrm{sw}}, S_{tk}\}$.
Compute the indicator function I_{tk} for the voice activity detection using (44.76) and (44.79).
Compute the conditional smoothed power spectrum $\tilde{S}_{tk}$ using (44.84) and (44.85), and update its running minimum: $\tilde{S}_{tk}^{\min} = \min\{\tilde{S}_{t-1,k}^{\min}, \tilde{S}_{tk}\};\ \tilde{S}_k^{\min\_\mathrm{sw}} = \min\{\tilde{S}_k^{\min\_\mathrm{sw}}, \tilde{S}_{tk}\}$.
Compute the a priori speech presence probability $\tilde{p}_{tk\lvert t-1}$ using (44.86) and (44.87), the speech presence probability $\tilde{p}_{tk}$ using (44.69), and the smoothing parameter $\tilde{\alpha}_{tk}$ using (44.68).
Update the noise spectrum estimate $\hat{\sigma}_{t+1,k}^2$ using (44.67) and (44.70).
Compute P_{tk}^{local} and P_{tk}^{global} using (44.32–44.34), and P_t^{frame} using the block diagram in Fig. 44.7.
Compute the a priori speech presence probability $\hat{p}_{tk\lvert t-1}$ using (44.37), and the speech presence probability $\hat{p}_{tk}$ using (44.69) by substituting $\tilde{p}_{tk\lvert t-1}$ with $\hat{p}_{tk\lvert t-1}$.
Compute the speech spectral estimate $\hat{X}_{tk}$ using (44.29).
Let $\ell = \ell + 1$.
If $\ell = V$
For all frequency bins k
Store $S_k^{\min\_\mathrm{sw}}$, set $S_{tk}^{\min}$ to the minimum of the last U stored values of $S_k^{\min\_\mathrm{sw}}$, and let $S_k^{\min\_\mathrm{sw}} = S_{tk}$.
Store $\tilde{S}_k^{\min\_\mathrm{sw}}$, set $\tilde{S}_{tk}^{\min}$ to the minimum of the last U stored values of $\tilde{S}_k^{\min\_\mathrm{sw}}$, and let $\tilde{S}_k^{\min\_\mathrm{sw}} = \tilde{S}_{tk}$.
Let $\ell = 0$.

a priori SNR estimation. The performance of the algorithm is demonstrated on speech signals degraded by various additive noise types.

The implementation of the speech enhancement algorithm is summarized in Table 44.5. For each time frame t we recursively estimate the STFT coefficients of the clean speech $\{X_{tk} | k = 1, \ldots, N/2\}$ from the noisy STFT coefficients $\{Y_{tk} | k = 1, \ldots, N/2\}$, where N is the length of the analysis window. We typically use a Hamming window of 32 ms length and a framing step of 8 ms (i. e., $N = 512$ and $M = 128$ for a sampling rate of 16 kHz). In the first frame ($t = 0$) we compute $\{Y_{0k} | k = 0, \ldots, N-1\}$ by applying the discrete Fourier transform to a short-time section of the noisy data

$$\boldsymbol{y}_0 = \left[y(0)h(0)\ \ y(1)h(1)\ \ \ldots\ \ y(N-1)h(N-1) \right]^{\mathrm{T}} ,$$

where $h(n)$ is the analysis window. In the following frames ($t > 0$), the section of noisy data is updated with M additional samples

$$\boldsymbol{y}_t = \left[y(tM)h(0)\quad y(1+tM)h(1) \quad \ldots \quad y(N-1+tM)h(N-1) \right]^{\mathrm{T}} ,$$

and subsequently $\{Y_{tk} | k = 0, \ldots, N-1\}$ is computed by applying the discrete Fourier transform to $\boldsymbol{y}_t$. Since the speech signal $x(n)$ is assumed to be real, once we estimate $\{X_{tk} | k = 1, \ldots, N/2\}$, the spectral coefficients for $N/2 < k \le N-1$ are obtained by $\hat{X}_{tk} = \hat{X}_{t,N-k}^*$, where * denotes complex conjugation. The DC component $\hat{X}_{to}$ is set to zero, and a sequence $\{\hat{x}_t(n) | n = 0, \ldots, N-1\}$ is obtained by applying the inverse discrete Fourier transform to $\{\hat{X}_{tk} | k = 0, \ldots, N-1\}$:

$$\hat{x}_t(n) = \frac{1}{N} \sum_{k=0}^{N-1} \hat{X}_{tk}\, \mathrm{e}^{\mathrm{i}\frac{2\pi}{N}nk} . \qquad (44.90)$$

Employing the weighted overlap-add method [44.37], we compute the following sequence

$$o_t(n) = \begin{cases} o_{t-1}(n+M) + N\tilde{h}(n)\hat{x}_t(n)\,, & \\ \qquad\qquad \text{for } 0 \le n \le N-M-1 & \\ N\tilde{h}(n)\hat{x}_t(n)\,, \quad \text{for } N-M \le n \le N-1\,, & \end{cases} \tag{44.91}$$

where $\tilde{h}(n)$ is the synthesis window. Then, according to (44.2), for each frame t, we obtain M additional samples of the enhanced speech signal:

$$\hat{x}(n+tM) = o_t(n)\,, \quad n = 0, \ldots, M-1\,. \tag{44.92}$$

The synthesis window $\tilde{h}(n)$ should satisfy the completeness condition [44.36]

$$\sum_t \tilde{h}(n-tM)h(n-tM) = \frac{1}{N} \quad \text{for all } n\,. \tag{44.93}$$

Given analysis and synthesis windows that satisfy (44.93), any signal $x(n) \in \ell_2(\mathbb{Z})$ can be perfectly reconstructed from its STFT coefficients X_{tk}. However, for $M < N$ (over-redundant STFT representation) and for a given analysis window $h(n)$, there might be an infinite number of solutions to (44.93). A reasonable choice of a synthesis window is the one with minimum energy [44.36, 58], given by

$$\tilde{h}(n) = \frac{h(n)}{N\sum_\ell h^2(n-\ell M)}\,. \tag{44.94}$$

The estimator for the a priori speech presence probability, $\tilde{p}_{tk|t-1}$ in (44.87), requires two iterations of time–frequency smoothing (S_{tk}, $\tilde{S}_{tk}$) and minimum tracking ($S_{tk}^{\min}$, $\tilde{S}_{tk}^{\min}$). The minimum tracking is implemented by the method proposed in [44.56, 59], which provides a flexible balance between the computational complexity and the update rate of the minima values. Accordingly, we divide the window of D samples into U subwindows of V samples ($UV = D$). Whenever V samples are read, the minimum of the current subwindow is determined and stored for later use. The overall minimum is obtained as the minimum of past samples within the current subwindow and the U previous subwindow minima. Typical values of D and V correspond to 960 ms and 120 ms, respectively. That is, for a framing step of 8 ms (i. e., $M = 128$ for a sampling rate of 16 kHz) we set $D = 120$, $V = 15$, and $U = 8$.

To demonstrate the performance of the speech enhancement algorithm, utterances are taken from the TIMIT database [44.38], degraded by various noise types from the Noisex92 database [44.60], and enhanced by the algorithm in Table 44.5. A clean utterance from a female speaker is shown in Fig. 44.12. The speech signal is sampled at 16 kHz and degraded by the various noise types, which include white Gaussian noise, car interior noise, F16 cockpit noise, and babble noise. Figure 44.13 shows the noisy speech signals with SNR of 5 dB. The corresponding enhanced speech signals are shown in Fig. 44.14.

The performance is evaluated by three objective quality measures and informal listening tests. The first quality measure is the segmental SNR (SegSNR), in dB, defined by [44.61]

$$\text{SegSNR} = \frac{1}{T}\sum_{t=0}^{T-1} \mathcal{C}(\text{SNR}_t)\,, \tag{44.95}$$

where T denotes the number of frames in the signal, and

$$\text{SNR}_t = 10\log_{10}\frac{\sum_{n=tM}^{tM+N-1} x^2(n)}{\sum_{n=tM}^{tM+N-1}[x(n)-\hat{x}(n)]^2} \tag{44.96}$$

represents the SNR in the t-th frame. The operator $\mathcal{C}$ confines the SNR at each frame to the perceptually meaningful range between 35 dB and −10 dB ($\mathcal{C}x \triangleq \min[\max(x, -10), 35]$). The operator $\mathcal{C}$ prevents the segmental SNR measure from being biased in either a positive or negative direction due to a few silent or unusually high-SNR frames, which do not contribute

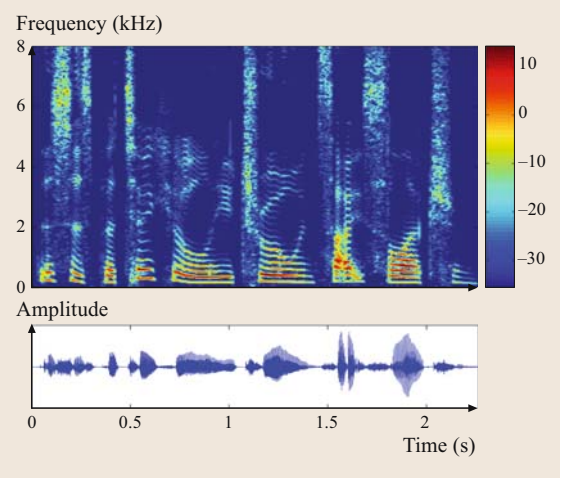


Fig. 44.12 Speech spectrogram and waveform of a clean speech signal: 'This is particularly true in site selection'

significantly to the overall speech quality [44.62, 63]. The second quality measure is log-spectral distortion (LSD), in dB, which is defined by

$$\mathrm{LSD} = \frac{1}{T}\sum_{t=0}^{T-1}\left[\frac{2}{N}\sum_{k=1}^{N/2}\left(\mathcal{L}X_{tk} - \mathcal{L}\hat{X}_{tk}\right)^2\right]^{\frac{1}{2}}, \tag{44.97}$$

where $\mathcal{L}X_{tk} \triangleq \max\{20\log_{10}|X_{tk}|, \delta\}$ is the log spectrum confined to about 50 dB dynamic range (that is, $\delta = \max_{tk}\{20\log_{10}|X_{tk}|\} - 50$). The third quality measure is the perceptual evaluation of speech quality (PESQ) score (ITU-T P.862).

The experimental results for the noisy and enhanced signals are given in the captions of Figs. 44.13 and 44.14. The improvement in SegSNR, reduction in LSD, and increase in PESQ scores are summarized in Table 44.6. Generally, the improvement in SegSNR and reduction in LSD are influenced by the variability of the noise characteristics in time and the initial SegSNR and LSD of the noisy signal. The faster the noise spectrum varies in time, the less reliable the noise spectrum estimator, and consequently the lower the quality gain that can be achieved by the speech enhancement system. Furthermore, the lower the SegSNR, respectively the LSD, for the noisy signal, the higher is the SegSNR improvement, respectively the lower is the LSD reduc-

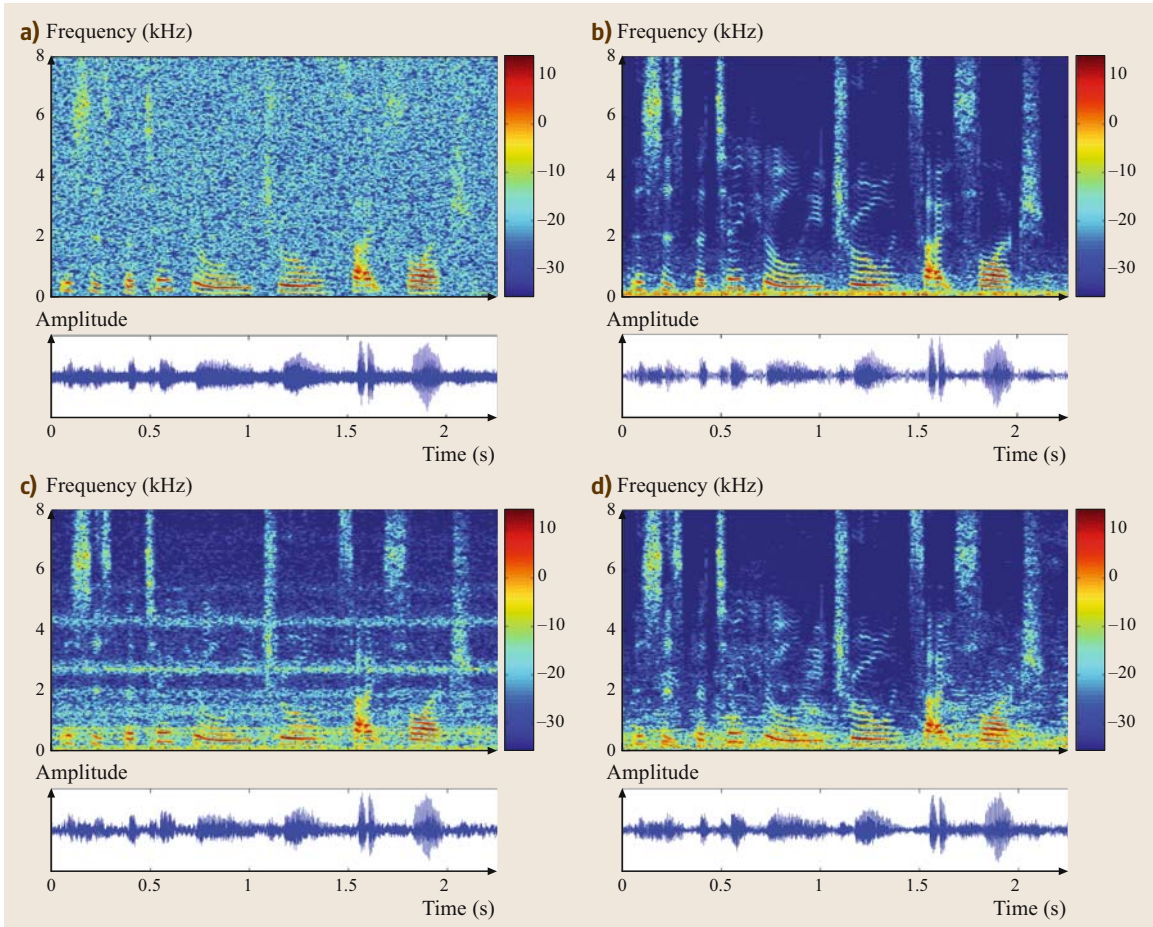


Fig. 44.13a–d Speech spectrograms and waveforms of the speech signal shown in Fig. 44.12 degraded by various noise types with SNR = 5 dB. **(a)** White Gaussian noise (SegSNR = −0.46 dB, LSD = 12.67 dB, PESQ = 1.74); **(b)** car interior noise (SegSNR = 0.30 dB, LSD = 3.48 dB, PESQ = 2.47); **(c)** F16 cockpit noise (SegSNR = −0.33 dB, LSD = 7.99 dB, PESQ = 1.76); **(d)** babble noise (SegSNR = 0.09 dB, LSD = 5.97 dB, PESQ = 1.87)

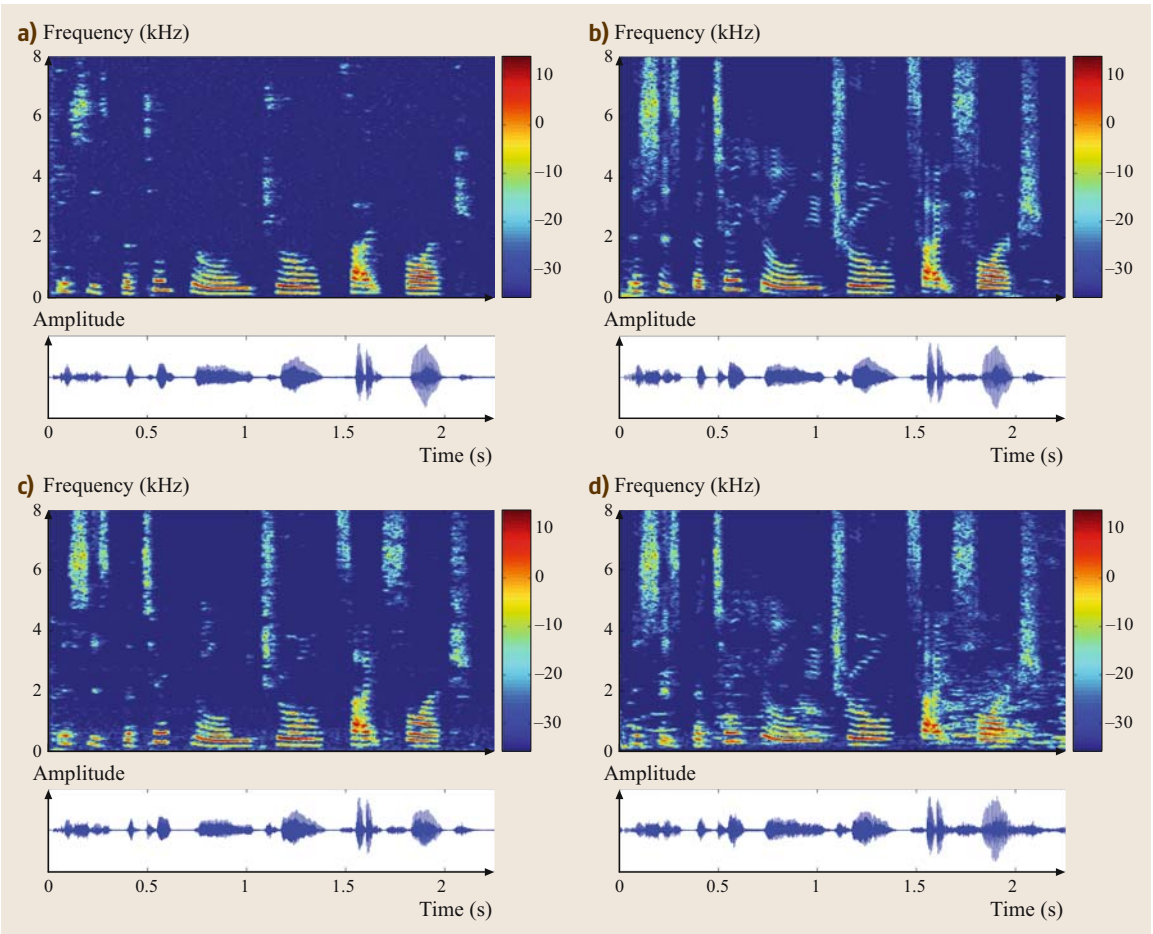


Fig. 44.14a–d Speech spectrograms and waveforms of the signals shown in Fig. 44.13 after enhancement with the algorithm in Table 44.5. **(a)** White Gaussian noise (SegSNR = 5.78 dB, LSD = 5.10 dB, PESQ = 2.34); **(b)** car interior noise (SegSNR = 9.52 dB, LSD = 2.67 dB, PESQ = 3.00); **(c)** F16 cockpit noise (SegSNR = 5.21 dB, LSD = 4.27 dB, PESQ = 2.29); **(d)** babble noise (SegSNR = 4.23 dB, LSD = 4.30 dB, PESQ = 2.13)

tion, that can be achieved by the speech enhancement system.

For car interior noise, most of the noise energy is concentrated in the lower frequencies. Therefore, the noise reduction is large in the low frequencies, and small in the high frequencies. Accordingly, in each frame, the total noise reduction is higher than that obtainable for the case of WGN. Since the SegSNR is mainly affected by the amount of noise reduction in each frame, the improvement in SegSNR is more significant for car interior noise. However, the reduction in LSD for the car interior noise is less substantial, since the initial LSD for the noisy signal is small.

The characteristics of babble noise vary more quickly in time when compared to the other noise

Table 44.6 Segmental SNR improvement, log-spectral distortion reduction and PESQ score improvement for various noise types, obtained by using the speech enhancement algorithm in Table 44.5

Noise type	SegSNR improvement	LSD reduction	PESQ score improvement
WGN	6.24	7.57	0.60
Car	9.22	0.81	0.53
F16	5.54	3.72	0.53
Babble	4.14	1.67	0.26

types. Therefore, the IMCRA noise spectrum estimator is least reliable for babble noise, and the speech enhancement performance is inferior to that achievable in slowly time-varying noise environments. Accordingly, the improvement in SegSNR is smaller for babble noise, when compared with the improvement achieved in more-stationary noise environment with the same initial SegSNR. Similarly, the reduction in LSD, and improvement in PESQ score, are smaller for babble noise, when compared with those achieved in more-stationary noise environment with the same initial LSD, respectively PESQ score, levels.

44.9 Selection of Spectral Enhancement Algorithms

In this section, we discuss some of the fundamental components that constitute a speech spectral enhancement system. Specifically, we address the choice of a statistical model, fidelity criterion, a priori SNR estimator, and noise spectrum estimator.

44.9.1 Choice of a Statistical Model and Fidelity Criterion

The Gaussian model underlies the design of many speech enhancement algorithms, e.g., [44.12, 17, 18, 42, 64–66]. This model is motivated by the central limit theorem, as each Fourier expansion coefficient is a weighted sum of random variables resulting from the random sequence [44.12]. When the span of correlation within the signal is sufficiently short compared to the size of the frames, the probability distribution function of the spectral coefficients asymptotically approaches Gaussian as the frame's size increases. The Gaussian approximation is in the central region of the Gaussian curve near the mean. However, the approximation can be very inaccurate in the tail regions away from the mean [44.67]. *Porter* and *Boll* [44.46] pointed out that a priori speech spectra do not have a Gaussian distribution, but gamma-like distribution. They proposed to compute the optimal estimator directly from the speech data, rather than from a parametric model of the speech statistics.

Martin [44.40] considered a gamma speech model, in which the real and imaginary parts of the clean speech spectral components are modeled as iid gamma random variables. He assumed that distinct spectral components are statistically independent, and derived MMSE estimators for the complex speech spectral coefficients under Gaussian and Laplacian noise modeling. He showed that, under Gaussian noise modeling, the gamma speech model yields a greater improvement in the segmental SNR than the Gaussian speech model. Under Laplacian noise modeling, the gamma speech model results in lower residual musical noise than the Gaussian speech model. *Martin* and *Breithaupt* [44.45] showed that when modeling the real and imaginary parts of the clean speech spectral components as Laplacian random variables, the MMSE estimators for the complex speech spectral coefficients have similar properties to those estimators derived under gamma modeling, but are easier to compute and implement.

Breithaupt and *Martin* [44.68] derived, using the same statistical modeling, MMSE estimators for the magnitude-squared spectral coefficients, and compared their performance to that obtained by using a Gaussian speech model. They showed that improvement in the segmental SNR comes at the expense of additional residual musical noise. *Lotter* and *Vary* [44.69] derived a maximum a posteriori (MAP) estimator for the speech spectral amplitude, based on a Gaussian noise model and a superGaussian speech model. They proposed a parametric pdf for the speech spectral amplitude, which approximates, with a proper choice of the parameters, the gamma and Laplacian densities. Compared with the MMSE spectral amplitude estimator of Ephraim-Malah, the MAP estimator with Laplacian speech modeling demonstrates improved noise reduction.

The Gaussian, gamma, and Laplacian models presented in Sect. 44.3 take into account the time correlation between successive speech spectral components. Spectral components in the STFT domain are assumed to be statistically correlated along the frequency axis, as well as along time trajectories, due to the finite length of the analysis frame in the STFT and the overlap between successive frames [44.15]. Experimental results of speech enhancement performance show [44.16, 43] that the appropriateness of the Gaussian, gamma, and Laplacian speech models are greatly affected by the particular choice of the a priori SNR estimator. When the a priori SNR is estimated by the decision-directed method, the gamma model is more advantageous than the Gaussian model. However, when the a priori SNR is estimated by the noncausal recursive estimation method, the Laplacian speech model yields a higher segmental SNR and a lower LSD than the other speech models,

while the level of residual musical noise is minimal when using a Gaussian speech model. Furthermore, the differences between the Gaussian, gamma, and Laplacian speech models are smaller when using the noncausal a priori estimators than when using the decision-directed method.

It is worthwhile noting that estimators that minimize the MSE distortion of the spectral amplitude or log-spectral amplitude are more suitable for speech enhancement than MMSE estimators. Moreover, it is difficult, or even impossible, to derive analytical expressions for MMSE-LSA estimators under gamma or Laplacian models. Therefore, the MMSE-LSA estimator derived under a Gaussian model is often preferred over the MMSE estimators derived under the other speech models [44.2].

44.9.2 Choice of an A Priori SNR Estimator

Ephraim and *Malah* [44.12, 70] proposed three different methods for the a priori SNR estimation. First, maximum-likelihood estimation, which relies on the assumption that the speech spectral variances are slowly time-varying parameters. This results in musical residual noise, which is annoying and disturbing to the perception of the enhanced signal. Second, decision-directed approach which is particularly useful when combined with the MMSE spectral, or log-spectral, magnitude estimators [44.12, 17, 47]. This results in perceptually colorless residual noise, but is heuristically motivated and its theoretical performance is unknown due to its highly nonlinear nature. Third, maximum a posteriori estimation, which relies on a first-order Markov model for generating a sequence of speech spectral variances. It involves a set of nonlinear equations, which are solved recursively by using the Viterbi algorithm. The computational complexity of the MAP estimator is relatively high, while it does not provide a significant improvement in the enhanced speech quality over the decision-directed estimator [44.70].

The decision-directed approach has become over the last two decades the most acceptable estimation method for the variances of the speech spectral coefficients. However, the parameters of the decision-directed estimator have to be determined by simulations and subjective listening tests for each particular setup of time–frequency transformation and speech enhancement algorithm. Furthermore, since the decision-directed approach is not supported by a statistical model, the parameters are not adapted to the speech components, but are set to specific values in advance. *Ephraim* and *Malah* recognized the limits of their variance estimation methods, and concluded that better speech enhancement performance may be obtained if the variance estimation could be improved [44.12, 70].

The causal estimator for the a priori SNR combines two steps, a *propagation* step and an *update* step, following the rational of Kalman filtering, to predict and update the estimate for the speech spectral variance recursively as new data arrive. The causal a priori SNR estimator is closely related to the decision-directed estimator of Ephraim and Malah. A special case of the causal estimator degenerates to a *decision-directed* estimator with a *time-varying frequency-dependent* weighting factor. The weighting factor is monotonically decreasing as a function of the instantaneous SNR, resulting effectively in a larger weighting factor during speech absence, and a smaller weighting factor during speech presence. This slightly reduces both the musical noise and the signal distortion. Nevertheless, the improvement in speech enhancement performance obtained by using the causal recursive over using the decision-directed method is not substantial. Therefore, if the delay between the enhanced speech and the noisy observation needs to be minimized, the decision-directed method is perhaps preferable due to its computational simplicity. However, in applications where a few-frames delay is tolerable, e.g., digital voice recording, surveillance, and speaker identification, the *noncausal* recursive estimation approach is more advantageous than the decision-directed approach.

The noncausal a priori SNR estimator employs future spectral measurements to predict the spectral variances of clean speech better. A comparison of the causal and noncausal estimators indicates that the differences are primarily noticeable during speech onset. The *causal* a priori SNR estimator, as well as the decision-directed estimator, cannot respond too quickly to an abrupt increase in the instantaneous SNR, since this necessarily implies an increase in the level of musical residual noise. In contrast, the *noncausal* estimator, having a few subsequent spectral measurements at hand, is capable of discriminating between speech onsets and noise irregularities. Experimental results show that the advantages of the noncausal estimator are particularly perceived during onsets of speech and noise only frames. Onsets of speech are better preserved, while a further reduction of musical noise is achieved [44.15, 53]. Furthermore, the differences between the Gaussian, gamma, and Laplacian speech models are smaller when using the noncausal recursive estimation approach than when using the decision-directed method [44.43].

44.9.3 Choice of a Noise Estimator

Traditional noise estimation methods are based on recursive averaging during sections that do not contain speech and holding the estimates during sections which contain speech. However, these methods generally require VADs and the update of the noise estimate is restricted to periods of speech absence. Additionally, VADs are difficult to tune and their reliability severely deteriorates for weak speech components and low input SNR [44.65, 71, 72]. Alternative techniques, based on histograms in the power spectral domain [44.55, 73, 74], are computationally expensive, require much more memory resources, and do not perform well in low-SNR conditions. Furthermore, the signal segments used for building the histograms are typically several hundred milliseconds long, and thus the update rate of the noise estimate is essentially moderate.

A useful noise estimation approach known as the minimum statistics [44.59] tracks the minima values of a smoothed power estimate of the noisy signal, and multiplies the result by a factor that compensates the bias. However, the variance of this noise estimate is about twice as large as the variance of a conventional noise estimator [44.59]. Moreover, this method may occasionally attenuate low-energy phonemes, particularly if the minimum search window is too short [44.75]. These limitations can be overcome, at the price of higher complexity, by adapting the smoothing parameter and the bias compensation factor in time and frequency [44.56].

A computationally efficient minimum tracking scheme is presented in [44.57]. Its main drawbacks are the slow update rate of the noise estimate in the case of a sudden rise in the noise energy level, and its tendency to cancel the signal [44.71]. Other closely related techniques are the *lower-energy envelope tracking* [44.55] and the *quantile-based* [44.76] estimation methods. Rather than picking the minima values of a smoothed periodogram, the noise is estimated based on a temporal quantile of a nonsmoothed periodogram of the noisy signal. Unfortunately, these methods suffer from the high computational complexity associated with the sorting operation, and the extra memory required for keeping past spectral power values.

The IMCRA noise estimator [44.54], presented in Sect. 44.7, combines the robustness of the minimum tracking with the simplicity of recursive averaging. Rather than employing a voice activity detector and restricting the update of the noise estimator to periods of speech absence, the smoothing parameter is adapted in time and frequency according to the speech presence probability. The noise estimate is thereby continuously updated even during weak speech activity. The estimator is controlled by the minima values of a smoothed periodogram of the noisy measurement. It combines conditions on both the instantaneous and local measured power, and provides a soft transition between speech absence and presence. This prevents an occasional increase in the noise estimate during speech activity. Furthermore, carrying out the smoothing and minimum tracking in two iterations allows larger smoothing windows and smaller minimum search windows, while reliably tracking the minima even during strong speech activity. This yields a reduced variance of the minima values and shorter delay when responding to a rising noise power, which eventually improves the tracking capability of the noise estimator. In nonstationary noise environments and under low-SNR conditions, the IMCRA approach is particularly useful [44.54]. It facilitates a lower estimation error, and when integrated into a speech enhancement system, yields improved speech quality and lower residual noise.

44.10 Conclusions

We have described statistical models for speech and noise signals in the STFT domain, and presented estimators for the speech spectral coefficients under speech presence uncertainty. The statistical models take into consideration the time correlation between successive spectral components of the speech signal. The spectral estimators involve estimation of the noise power spectrum, calculation of the speech presence probability, and evaluation of the a priori SNR under speech presence uncertainty. We discussed the behavior of the MMSE-LSA spectral gain function and its advantage for the mechanism that counters the musical noise phenomenon. Local bursts of the a posteriori SNR during noise-only frames are pulled down to the average noise level, thus avoiding local buildup of noise whenever it exceeds its average characteristics. The estimator for the a priori speech presence probability exploits the strong correlation of speech presence in neighboring frequency bins of consecutive frames, which enables further attenuation of noise components while avoid-

ing clipping of speech onsets and misdetection of weak speech tails.

We have presented estimators for the a priori SNR under speech presence uncertainty, and showed that a special case of a causal recursive estimator degenerates to a decision-directed estimator with a time-varying frequency-dependent weighting factor. Furthermore, in applications where a delay of a few short-term frames between the enhanced speech and the noisy observation is tolerable, a noncausal estimation approach may produce less signal distortion and less musical residual noise than a causal estimation approach.

We described the IMCRA approach for the noise power spectrum estimation, and provided a detailed example of a speech enhancement algorithm. We showed that the improvement in SegSNR and reduction in LSD are influenced by the variability of the noise characteristics in time and the initial SegSNR and LSD of the noisy signal. The faster the noise spectrum varies in time, the less reliable the noise spectrum estimator, and consequently the lower the quality gain that can be achieved by the speech enhancement system. Furthermore, the lower the initial quality of the noisy signal, the larger the improvement that can be achieved by the speech enhancement system.

References

44.1 J. Benesty, S. Makino, J. Chen (Eds.): *Speech Enhancement* (Springer, Berlin, Heidelberg 2005)

44.2 Y. Ephraim, I. Cohen: Recent advancements in speech enhancement. In: *The Electrical Engineering Handbook, Circuits, Signals, and Speech and Image Processing*, 3rd edn., ed. by R.C. Dorf (CRC, Boca Raton 2006) pp. 15–12–15–26, Chap. 15

44.3 Y. Ephraim, H. Lev-Ari, W.J.J. Roberts: A brief survey of speech enhancement. In: *The Electronic Handbook*, 2nd edn. (CRC-Press, Boca Raton 2005)

44.4 S.F. Boll: Suppression of acoustic noise in speech using spectral subtraction, IEEE Trans. Acoust. Speech Signal Process. **27**(2), 113–120 (1979)

44.5 J.S. Lim, A.V. Oppenheim: Enhancement and bandwidth compression of noisy speech, Proc. IEEE **67**(12), 1586–1604 (1979)

44.6 M. Berouti, R. Schwartz, J. Makhoul: Enhancement of speech corrupted by acoustic noise, Proc. 4th ICASSP **79**, 208–211 (1979)

44.7 Z. Goh, K.-C. Tan, T.G. Tan: Postprocessing method for suppressing musical noise generated by spectral subtraction, IEEE Trans. Speech Audio Process. **6**(3), 287–292 (1998)

44.8 B.L. Sim, Y.C. Tong, J.S. Chang, C.T. Tan: A parametric formulation of the generalized spectral subtraction method, IEEE Trans. Speech Audio Process. **6**(4), 328–337 (1998)

44.9 H. Gustafsson, S.E. Nordholm, I. Claesson: Spectral subtraction using reduced delay convolution and adaptive averaging, IEEE Trans. Speech Audio Process. **9**(8), 799–807 (2001)

44.10 D.E. Tsoukalas, J.N. Mourjopoulos, G. Kokkinakis: Speech enhancement based on audible noise suppression, IEEE Trans. Speech Audio Process. **5**(6), 497–514 (1997)

44.11 N. Virag: Single channel speech enhancement based on masking properties of the human auditory system, IEEE Trans. Speech Audio Process. **7**(2), 126–137 (1999)

44.12 Y. Ephraim, D. Malah: Speech enhancement using a minimum mean-square error short-time spectral amplitude estimator, IEEE Trans. Acoust. Speech Signal Process. **ASSP-32**(6), 1109–1121 (1984)

44.13 D. Burshtein, S. Gannot: Speech enhancement using a mixture-maximum model, IEEE Trans. Speech Audio Process. **10**(6), 341–351 (2002)

44.14 R. Martin: Speech enhancement based on minimum mean-square error estimation and supergaussian priors, IEEE Trans. Speech Audio Process. **13**(5), 845–856 (2005)

44.15 I. Cohen: Relaxed statistical model for speech enhancement and a priori SNR estimation, IEEE Trans. Speech Audio Process. **13**(5), 870–881 (2005)

44.16 I. Cohen: Speech spectral modeling and enhancement based on autoregressive conditional heteroscedasticity models, Signal Process. **86**(4), 698–709 (2006)

44.17 Y. Ephraim, D. Malah: Speech enhancement using a minimum mean-square error log-spectral amplitude estimator, IEEE Trans. Acoust. Speech Signal Process. **ASSP-33**(2), 443–445 (1985)

44.18 P.J. Wolfe, S.J. Godsill: Efficient alternatives to the Ephraim and Malah suppression rule for audio signal enhancement, Special Issue EURASIP JASP Digital Audio Multim. Commun. **2003**(10), 1043–1051 (2003)

44.19 P.C. Loizou: Speech enhancement based on perceptually motivated bayesian estimators of the magnitude spectrum, IEEE Trans. Speech Audio Process. **13**(5), 857–869 (2005)

44.20 B.H. Juang, L.R. Rabiner: Mixture autoregressive hidden Markov models for speech signals, IEEE Trans. Acoust. Speech Signal Process. **ASSP-33**(6), 1404–1413 (1985)

44.21 Y. Ephraim: Statistical-model-based speech enhancement systems, Proc. IEEE **80**(10), 1526–1555 (1992)

44.22 H. Sheikhzadeh, L. Deng: Waveform-based speech recognition using hidden filter models: Parameter selection and sensitivity to power normalization, IEEE Trans. Speech Audio Process. **2**, 80–91 (1994)

44.23 Y. Ephraim, N. Merhav: Hidden Markov processes, IEEE Trans. Inform. Theory **48**(6), 1518–1568 (2002)

44.24 C.J. Wellekens: Explicit time correlations in hidden Markov models for speech recognition, Proc. 12th ICASSP **87**, 384–386 (1987)

44.25 H. Sameti, H. Sheikhzadeh, L. Deng, R.L. Brennan: HMM-based strategies for enhancement of speech signals embedded in nonstationary noise, IEEE Trans. Speech Audio Process. **6**(5), 445–455 (1998)

44.26 L.R. Rabiner, B.-H. Juang: *Fundamentals of Speech Recognition* (Prentice-Hall, Upper Saddle River 1993)

44.27 F. Jelinek: *Statistical Methods for Speech Recognition* (MIT Press, Cambridge 1998)

44.28 Y. Ephraim, H.L.V. Trees: A signal subspace approach for speech enhancement, IEEE Trans. Speech Audio Process. **3**(4), 251–266 (1995)

44.29 F. Asano, S. Hayamizu, T. Yamada, S. Nakamura: Speech enhancement based on the subspace method, IEEE Trans. Speech Audio Process. **8**(5), 497–507 (2000)

44.30 U. Mittal, N. Phamdo: Signal/noise KLT based approach for enhancing speech degraded by colored noise, IEEE Trans. Speech Audio Process. **8**(2), 159–167 (2000)

44.31 Y. Hu, P.C. Loizou: A generalized subspace approach for enhancing speech corrupted by colored noise, IEEE Trans. Speech Audio Process. **11**(4), 334–341 (2003)

44.32 S.H. Jensen, P.C. Hansen, S.D. Hansen, J.A. Sørensen: Reduction of broad-band noise in speech by truncated QSVD, IEEE Trans. Speech Audio Process. **3**(6), 439–448 (1995)

44.33 S. Doclo, M. Moonen: GSVD-based optimal filtering for single and multimicrophone speech enhancement, IEEE Trans. Signal Process. **50**(9), 2230–2244 (2002)

44.34 F. Jabloun, B. Champagne: Incorporating the human hearing properties in the signal subspace approach for speech enhancement, IEEE Trans. Speech Audio Process. **11**(6), 700–708 (2003)

44.35 Y. Hu, P.C. Loizou: A perceptually motivated approach for speech enhancement, IEEE Trans. Speech Audio Process. **11**(5), 457–465 (2003)

44.36 J. Wexler, S. Raz: Discrete Gabor expansions, Speech Process. **21**(3), 207–220 (1990)

44.37 R.E. Crochiere, L.R. Rabiner: *Multirate Digital Signal Processing* (Prentice-Hall, Englewood Cliffs 1983)

44.38 J.S. Garofolo: *Getting Started with the DARPA TIMIT CD-ROM: An Acoustic Phonetic Continuous Speech Database* (NIST, Gaithersburg 1988)

44.39 A. Stuart, J.K. Ord: *Kendall's Advanced Theory of Statistics*, Vol. 1, 6th edn. (Arnold, London 1994)

44.40 R. Martin: Speech enhancement using MMSE short time spectral estimation with Gamma distributed speech priors, Proc. 27th ICASSP **02**, 253–256 (2002)

44.41 I. Cohen: Modeling speech signals in the time-frequency domain using GARCH, Signal Process. **84**(12), 2453–2459 (2004)

44.42 I. Cohen, B. Berdugo: Speech enhancement for non-stationary noise environments, Signal Process. **81**(11), 2403–2418 (2001)

44.43 I. Cohen: Speech enhancement using supergaussian speech models and noncausal a priori SNR estimation, Speech Commun. **47**(3), 336–350 (2005)

44.44 I.S. Gradshteyn, I.M. Ryzhik: *Table of Integrals, Series, and Products*, 4th edn. (Academic Press, New York 1980)

44.45 R. Martin, C. Breithaupt: Speech enhancement in the DFT domain using Laplacian speech priors. In: *Proc. 8th Int. Workshop on Acoustic Echo and Noise Control* (Kyoto, Japan 2003) pp. 87–90

44.46 J. Porter, S. Boll: Optimal estimators for spectral restoration of noisy speech, Proc. ICASSP **84**, 18A.2.1–18A.2.4 (1984)

44.47 O. Cappé: Elimination of the musical noise phenomenon with the Ephraim and Malah noise suppressor, IEEE Trans. Acoust. Speech Signal Process. **2**(2), 345–349 (1994)

44.48 P. Scalart, J. Vieira-Filho: Speech enhancement based on a priori signal to noise estimation, Proc. 21th ICASSP **96**, 629–632 (1996)

44.49 D. Malah, R.V. Cox, A.J. Accardi: Tracking speech-presence uncertainty to improve speech enhancement in non-stationary noise environments, Proc. 24th ICASSP **99**, 789–792 (1999)

44.50 I. Cohen: On speech enhancement under signal presence uncertainty, Proc. 26th ICASSP **2001**, 167–170 (2001)

44.51 I.Y. Soon, S.N. Koh, C.K. Yeo: Improved noise suppression filter using self-adaptive estimator of probability of speech absence, Signal Process. **75**(2), 151–159 (1999)

44.52 M. Marzinzik: *Noise reduction schemes for digital hearing aids and their use for the hearing impaired*, Ph.D. Thesis (Oldenburg University, Oldenburg 2000)

44.53 I. Cohen: Speech enhancement using a noncausal a priori SNR estimator, IEEE Signal Process. Lett. **11**(9), 725–728 (2004)

44.54 I. Cohen: Noise spectrum estimation in adverse environments: Improved minima controlled recursive averaging, IEEE Trans. Speech Audio Process. **11**(5), 466–475 (2003)

44.55 C. Ris, S. Dupont: Assessing local noise level estimation methods: Application to noise robust ASR, Speech Commun. **34**(1–2), 141–158 (2001)

44.56 R. Martin: Noise power spectral density estimation based on optimal smoothing and minimum statis-

tics, IEEE Trans. Speech Audio Process. **9**(5), 504–512 (2001)
44.57 G. Doblinger: Computationally efficient speech enhancement by spectral minima tracking in subbands, Proc. 4th Eurospeech **95**, 1513–1516 (1995)
44.58 S. Qian, D. Chen: Discrete Gabor transform, IEEE Trans. Signal Process. **41**(7), 2429–2438 (1993)
44.59 R. Martin: Spectral subtraction based on minimum statistics, Proc. 7th EUSIPCO **94**, 1182–1185 (1994)
44.60 A. Varga, H.J.M. Steeneken: Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems, Speech Commun. **12**(3), 247–251 (1993)
44.61 S.R. Quackenbush, T.P. Barnwell, M.A. Clements: *Objective Measures of Speech Quality* (Prentice-Hall, Englewood Cliffs 1988)
44.62 J.R. Deller, J.H.L. Hansen, J.G. Proakis: *Discrete-Time Processing of Speech Signals*, 2nd edn. (IEEE, New York 2000)
44.63 P.E. Papamichalis: *Practical Approaches to Speech Coding* (Prentice-Hall, Englewood Cliffs 1987)
44.64 A.J. Accardi, R.V. Cox: A modular approach to speech enhancement with an application to speech coding, Proc. 24th ICASSP **99**, 201–204 (1999)
44.65 J. Sohn, N.S. Kim, W. Sung: A statistical model-based voice activity detector, IEEE Signal Process. Lett. **6**(1), 1–3 (1999)
44.66 T. Lotter, C. Benien, P. Vary: Multichannel speech enhancement using bayesian spectral amplitude estimation, Proc. 28th ICASSP **03**, 832–835 (2003)
44.67 J.W.B. Davenport: *Probability and Random Processes: An Introduction for Applied Scientists and Engineers* (McGraw-Hill, New York 1970)
44.68 C. Breithaupt, R. Martin: MMSE estimation of magnitude-squared DFT coefficients with supergaussian priors, Proc. 28th ICASSP **03**, 896–899 (2003)
44.69 T. Lotter, P. Vary: Noise reduction by maximum a posteriori spectral amplitude estimation with supergaussian speech modeling. In: *Proc. 8th Internat. Workshop on Acoustic Echo and Noise Control* (2003) pp. 83–86
44.70 Y. Ephraim, D. Malah: *Signal to Noise Ratio Estimation for Enhancing Speech Using the Viterbi Algorithm*, Tech. Rep. EE PUB 489 (Technion – Israel Institute of Technology, Haifa 1984)
44.71 J. Meyer, K.U. Simmer, K.D. Kammeyer: Comparison of one- and two-channel noise-estimation techniques, Proc. 5th IWAENC **97**, 137–145 (1997)
44.72 B.L. McKinley, G.H. Whipple: Model based speech pause detection, Proc. 22th ICASSP **97**, 1179–1182 (1997)
44.73 R.J. McAulay, M.L. Malpass: Speech enhancement using a soft-decision noise suppression filter, IEEE Trans. Acoust. Speech Signal Process. **ASSP-28**(2), 137–145 (1980)
44.74 H.G. Hirsch, C. Ehrlicher: Noise estimation techniques for robust speech recognition, Proc. 20th ICASSP **95**, 153–156 (1995)
44.75 I. Cohen, B. Berdugo: Speech enhancement for non-stationary noise environments, Signal Process. **81**(11), 2403–2418 (2001)
44.76 V. Stahl, A. Fischer, R. Bippus: Quantile based noise estimation for spectral subtraction and Wiener filtering, Proc. 25th ICASSP **2000**, 1875–1878 (2000)

52. Convolutive Blind Source Separation Methods

M. S. Pedersen, J. Larsen, U. Kjems, L. C. Parra

In this chapter, we provide an overview of existing algorithms for blind source separation of convolutive audio mixtures. We provide a taxonomy in which many of the existing algorithms can be organized and present published results from those algorithms that have been applied to real-world audio separation tasks.

During the past decades, much attention has been given to the separation of mixed sources, in particular for the *blind* case where both the sources and the mixing process are unknown and only recordings of the mixtures are available. In several situations it is desirable to recover all sources from the recorded mixtures, or at least to segregate a particular source. Furthermore, it may be useful to identify the mixing process itself to reveal information about the physical mixing system.

In some simple mixing models each recording consists of a sum of differently weighted source signals. However, in many real-world applications, such as in acoustics, the mixing process is more complex. In such systems, the mixtures are weighted and delayed, and each source contributes to the sum with multiple delays corresponding to the multiple paths by which an acoustic signal propagates to a microphone. Such filtered sums of different sources are called convolutive mixtures. Depending on the situation, the filters may consist of a few delay elements, as in radio communications, or up to several thousand delay elements as in acoustics. In these situations the sources are the desired signals, yet only the recordings of the mixed sources are available and the mixing process is unknown.

There are multiple potential applications of convolutive blind source separation. In acoustics different sound sources are recorded simultaneously with possibly multiple microphones. These sources may be speech or music, or underwater signals recorded in passive sonar [52.1]. In radio communications, antenna arrays receive mixtures of different communication signals [52.2,3]. Source separation has also been applied to astronomical data or satellite images [52.4]. Finally, convolutive models have been used to interpret functional brain imaging data and biopotentials [52.5–8].

This chapter considers the problem of separating linear convolutive mixtures focusing in particular on acoustic mixtures. The *cocktail-party problem* has come to characterize the task of recovering speech in a room of simultaneous and independent speakers [52.9, 10]. Con-

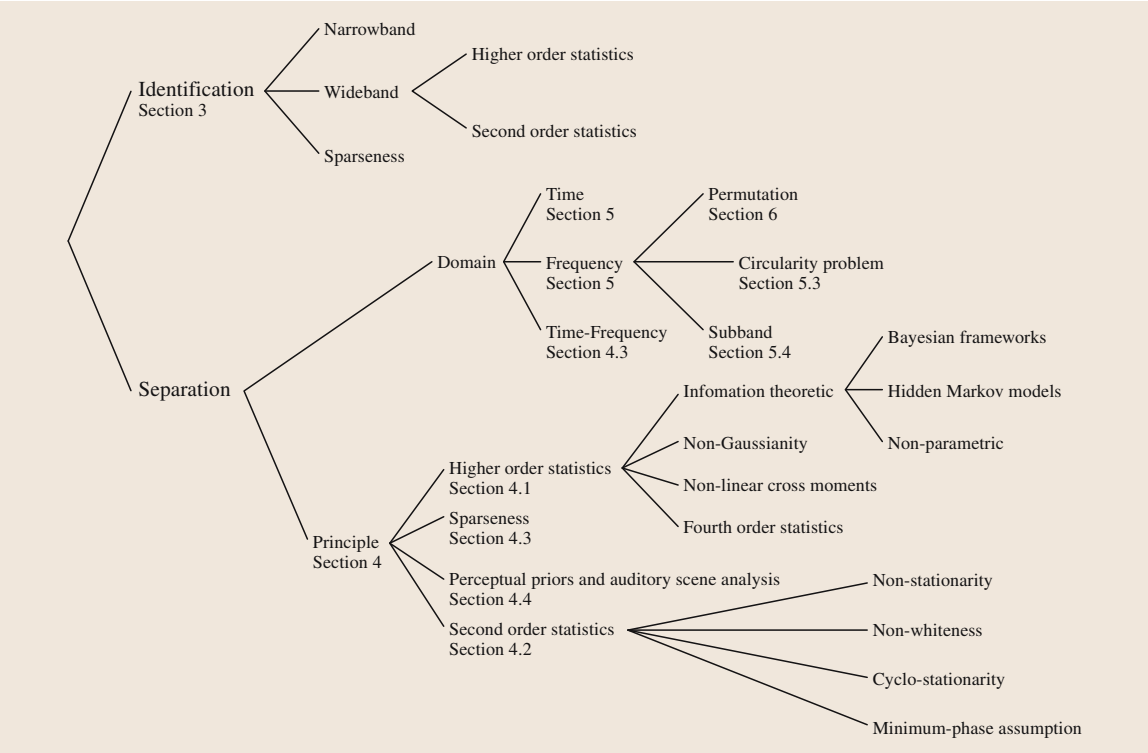


Fig. 52.1 Overview of important areas within blind separation of convolutive sources

volutive blind source separation (BSS) has often been proposed as a possible solution to this problem as it carries the promise to recover the sources exactly. The theory on linear noise-free systems establishes that a system with multiple inputs (sources) and multiple output (sensors) can be inverted under some reasonable assumptions with appropriately chosen multidimensional filters [52.11]. The challenge lies in finding these convolution filters.

There are already a number of partial reviews available on this topic [52.12–22]. The purpose of this chapter is to provide a complete survey of convolutive BSS and identify a taxonomy that can organize the large number of available algorithms. This may help practitioners and researchers new to the area of convolutive source separation obtain a complete overview of the field. Hopefully those with more experience in the field can identify useful tools, or find inspiration for new algorithms. Figure 52.1 provides an overview of the different topics within convolutive BSS and in which section they are covered. An overview of published results is given in Sect. 52.7.

52.1 The Mixing Model

First we introduce the basic model of convolutive mixtures. At the discrete time index t, a mixture of N source signals $\boldsymbol{s}(t) = [s_1(t), \ldots, s_N(t)]^{\mathrm{T}}$ are received at an array of M sensors. The received signals are denoted by $\boldsymbol{x}(t) = [x_1(t), \ldots, x_M(t)]^{\mathrm{T}}$. In many real-world applications the sources are said to be *convolutively* (or dynamically) mixed. The convolutive model introduces the following relation between the m-th mixed signal, the original source signals, and some additive sensor noise $v_m(t)$:

$$x_m(t) = \sum_{n=1}^{N} \sum_{k=0}^{K-1} a_{mnk} s_n(t-k) + v_m(t) \qquad (52.1)$$

The mixed signal is a linear mixture of filtered versions of each of the source signals, and a_{mnk} represent the corresponding mixing filter coefficients. In practice, these coefficients may also change in time, but for simplicity the mixing model is often assumed stationary. In theory the filters may be of infinite length, which may be implemented as infinite impulse response (IIR) systems, however, in practice it is sufficient to assume $K < \infty$. In matrix form the convolutive model can be written:

$$\boldsymbol{x}(t) = \sum_{k=0}^{K-1} \boldsymbol{A}_k \boldsymbol{s}(t-k) + \boldsymbol{v}(t) \,, \tag{52.2}$$

where $\boldsymbol{A}_k$ is an $M \times N$ matrix that contains the k-th filter coefficients. $\boldsymbol{v}(t)$ is the $M \times 1$ noise vector. In the z-domain the convolutive mixture (52.2) can be written:

$$\boldsymbol{X}(z) = \boldsymbol{A}(z)\boldsymbol{S}(z) + \boldsymbol{V}(z) \,, \tag{52.3}$$

where $\boldsymbol{A}(z)$ is a matrix with finite impulse response (FIR) polynomials in each entry [52.23].

52.1.1 Special Cases

There are some special cases of the convolutive mixture which simplify (52.2).

Instantaneous Mixing Model

Assuming that all the signals arrive at the sensors at the same time without being filtered, the convolutive mixture model (52.2) simplifies to

$$\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{s}(t) + \boldsymbol{v}(t) \,. \tag{52.4}$$

This model is known as the *instantaneous* or delayless (linear) mixture model. Here, $\boldsymbol{A} = \boldsymbol{A}_0$, is an $M \times N$ matrix containing the mixing coefficients. Many algorithms have been developed to solve the instantaneous mixture problem, see e.g., [52.17, 24].

Delayed Sources

Assuming a reverberation-free environment with propagation delays the mixing model can be simplified to

$$x_m(t) = \sum_{n=1}^{N} a_{mn} s_n(t - k_{mn}) + v_m(t) \,, \tag{52.5}$$

where k_{mn} is the propagation delay between source n and sensor m.

Noise Free

In the derivation of many algorithms, the convolutive model (52.2) is assumed to be noise-free, i. e.,

$$\boldsymbol{x}(t) = \sum_{k=0}^{K-1} \boldsymbol{A}_k \boldsymbol{s}(t-k) \,. \tag{52.6}$$

Over- and Underdetermined Sources

Often it is assumed that the number of sensors equals (or exceeds) the number of sources in which case linear methods may suffice to invert the linear mixing. However, if the number of sources exceeds the number of sensors the problem is underdetermined, and even under perfect knowledge of the mixing system linear methods will not be able to recover the sources.

52.1.2 Convolutive Model in the Frequency Domain

The convolutive mixing process (52.2) can be simplified by transforming the mixtures into the frequency domain. The linear convolution in the time domain can be written in the frequency domain as separate multiplications for each frequency:

$$\boldsymbol{X}(\omega) = \boldsymbol{A}(\omega)\boldsymbol{S}(\omega) + \boldsymbol{V}(\omega) \,. \tag{52.7}$$

At each frequency, $\omega = 2\pi f$, $\boldsymbol{A}(\omega)$ is a complex $M \times N$ matrix, $\boldsymbol{X}(\omega)$ and $\boldsymbol{V}(\omega)$ are complex $M \times 1$ vectors, and similarly $\boldsymbol{S}(\omega)$ is a complex $N \times 1$ vector. The frequency transformation is typically computed using a discrete Fourier transform (DFT) within a time frame of length T starting at time t:

$$\boldsymbol{X}(\omega, t) = \mathrm{DFT}([\boldsymbol{x}(t), \cdots, \boldsymbol{x}(t+T-1)]) \,, \tag{52.8}$$

and correspondingly for $\boldsymbol{S}(\omega, t)$ and $\boldsymbol{V}(\omega, t)$. Often a windowed discrete Fourier transform is used:

$$\boldsymbol{X}(\omega, t) = \sum_{\tau=0}^{T-1} w(\tau)\boldsymbol{x}(t+\tau)\mathrm{e}^{-i\omega\tau/T} \,, \tag{52.9}$$

where the window function $w(\tau)$ is chosen to minimize band overlap due to the limited temporal aperture. By using the fast Fourier transform (FFT) convolutions can be implemented efficiently in the discrete Fourier domain, which is important in acoustics as it often requires long time-domain filters.

52.1.3 Block-Based Model

Instead of modeling individual samples at time t one can also consider a block consisting of T samples. The equations for such a block can be written as:

$$\begin{aligned}
\boldsymbol{x}(t) &= \boldsymbol{A}_0\boldsymbol{s}(t)+\cdots+\boldsymbol{A}_{K-1}\boldsymbol{s}(t-K+1)\,,\\
\boldsymbol{x}(t-1) &= \boldsymbol{A}_0\boldsymbol{s}(t-1)+\cdots+\boldsymbol{A}_{K-1}\boldsymbol{s}(t-K)\,,\\
\boldsymbol{x}(t-2) &= \boldsymbol{A}_0\boldsymbol{s}(t-2)+\cdots+\boldsymbol{A}_{K-1}\boldsymbol{s}(t-K-1)\,,\\
&\vdots \qquad .
\end{aligned}$$

The M-dimensional output sequence can be written as an $MT\times 1$ vector:

$$\hat{\boldsymbol{x}}(t) = [\boldsymbol{x}^{\mathrm{T}}(t), \boldsymbol{x}^{\mathrm{T}}(t-1), \cdots, \boldsymbol{x}^{\mathrm{T}}(t-T+1)]^{\mathrm{T}}\,, \tag{52.10}$$

where $\boldsymbol{x}^{\mathrm{T}}(t) = [x_1(t), \cdots, x_M(t)]$. Similarly, the N-dimensional input sequence can be written as an $N(T+K-1)\times 1$ vector:

$$\hat{\boldsymbol{s}}(t) = [\boldsymbol{s}^{\mathrm{T}}(t), \boldsymbol{s}^{\mathrm{T}}(t-1), \cdots, \boldsymbol{s}^{\mathrm{T}}(t-T-K+2)]^{\mathrm{T}} \tag{52.11}$$

From this the convolutive mixture can be expressed formally as:

$$\hat{\boldsymbol{x}}(t) = \hat{\boldsymbol{A}}\hat{\boldsymbol{s}}(t)+\hat{\boldsymbol{v}}(t), \tag{52.12}$$

where $\hat{\boldsymbol{A}}$ has the following form:

$$\hat{\boldsymbol{A}} = \begin{pmatrix} \boldsymbol{A}_0 & \cdots & \boldsymbol{A}_{K-1} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \ddots & \ddots & \ddots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{A}_0 & \cdots & \boldsymbol{A}_{K-1} \end{pmatrix}. \tag{52.13}$$

The block-Toeplitz matrix $\hat{\boldsymbol{A}}$ has dimensions $MT\times N(T+K-1)$. On the surface, (52.12) has the same structure as an instantaneous mixture given in (52.4), and the dimensionality has increased by a factor T. However, the models differ considerably as the elements within $\hat{\boldsymbol{A}}$ and $\hat{\boldsymbol{s}}(t)$ are now coupled in a rather specific way.

The majority of the work in convolutive source separation assumes a mixing model with a finite impulse response as in (52.2). A notable exception is the work by Cichocki, which also considers an autoregressive (AR) component as part of the mixing model [52.18]. The autoregressive moving-average (ARMA) mixing system proposed there is equivalent to a first-order Kalman filter with an infinite impulse response.

52.2 The Separation Model

The objective of blind source separation is to find an estimate $\boldsymbol{y}(t)$ that is a model of the original source signals $\boldsymbol{s}(t)$. For this, it may not be necessary to identify the mixing filters $\boldsymbol{A}_k$ explicitly. Instead, it is often sufficient to estimate separation filters $\boldsymbol{W}_l$ that remove the cross-talk introduced by the mixing process. These separation filters may have a feed-back structure with an infinite impulse response, or may have a finite impulse response expressed as feedforward structure.

52.2.1 Feedforward Structure

An FIR separation system is given by

$$y_n(t) = \sum_{m=1}^{M}\sum_{l=0}^{L-1} w_{nml}x_m(t-l) \tag{52.14}$$

or in matrix form

$$\boldsymbol{y}(t) = \sum_{l=0}^{L-1} \boldsymbol{W}_l\boldsymbol{x}(t-l)\,. \tag{52.15}$$

As with the mixing process, the separation system can be expressed in the z-domain as

$$\boldsymbol{Y}(z) = \boldsymbol{W}(z)\boldsymbol{X}(z)\,, \tag{52.16}$$

and can also be expressed in block-Toeplitz form with the corresponding definitions for $\hat{\boldsymbol{y}}(t)$ and $\hat{\boldsymbol{W}}$ [52.25]:

$$\hat{\boldsymbol{y}}(t) = \hat{\boldsymbol{W}}\hat{\boldsymbol{x}}(t)\,. \tag{52.17}$$

Table 52.1 summarizes the mixing and separation equations in the different domains.

52.2.2 Relation Between Source and Separated Signals

The goal in source separation is not necessarily to recover identical copies of the original sources. Instead, the aim is to recover model sources without interferences from other sources, i. e., each separated signal $y_n(t)$ should contain signals originating from a single source only (Fig. 52.3). Therefore, each model source signal can be a filtered version of the original source signals, i. e.,

$$\boldsymbol{Y}(z) = \boldsymbol{W}(z)\boldsymbol{A}(z)\boldsymbol{S}(z) = \boldsymbol{G}(z)\boldsymbol{S}(z)\,, \tag{52.18}$$

Table 52.1 The convolutive mixing equation and its corresponding separation equation for different domains in which blind source separation algorithms have been derived

	Mixing process	Separation model
Time	$x_m(t)=\sum_{n=1}^{N}\sum_{k=0}^{K-1} a_{mnk}s_n(t-k)+v_m(t)$ $\boldsymbol{x}(t)=\sum_{k=0}^{K-1}\boldsymbol{A}_k\boldsymbol{s}(t-k)+\boldsymbol{v}(t)$	$y_n(t)=\sum_{m=1}^{M}\sum_{l=0}^{L-1} w_{nml}x_m(t-l)$ $\boldsymbol{y}(t)=\sum_{l=0}^{L-1}\boldsymbol{W}_l\boldsymbol{x}(t-l)$
z-domain	$\boldsymbol{X}(z)=\boldsymbol{A}(z)\boldsymbol{S}(z)+\boldsymbol{V}(z)$	$\boldsymbol{Y}(z)=\boldsymbol{W}(z)\boldsymbol{X}(z)$
Frequency domain	$\boldsymbol{X}(\omega)=\boldsymbol{A}(\omega)\boldsymbol{S}(\omega)+\boldsymbol{V}(\omega)$	$\boldsymbol{Y}(\omega)=\boldsymbol{W}(\omega)\boldsymbol{X}(\omega)$
Block-Toeplitz form	$\hat{\boldsymbol{x}}(t)=\hat{\boldsymbol{A}}\hat{\boldsymbol{s}}(t)$	$\hat{\boldsymbol{y}}(t)=\hat{\boldsymbol{W}}\hat{\boldsymbol{x}}(t)$

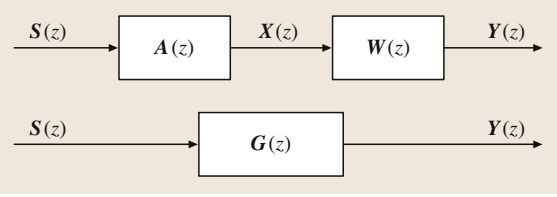


Fig. 52.2 The source signals $\boldsymbol{Y}(z)$ are mixed with the mixing filter $\boldsymbol{A}(z)$. An estimate of the source signals is obtained through an unmixing process, where the received signals $\boldsymbol{X}(z)$ are unmixed with the filter $\boldsymbol{W}(z)$. Each estimated source signal is then a filtered version of the original source, i. e., $\boldsymbol{G}(z)=\boldsymbol{W}(z)\boldsymbol{A}(z)$. Note that the mixing and the unmixing filters do not necessarily have to be of the same order

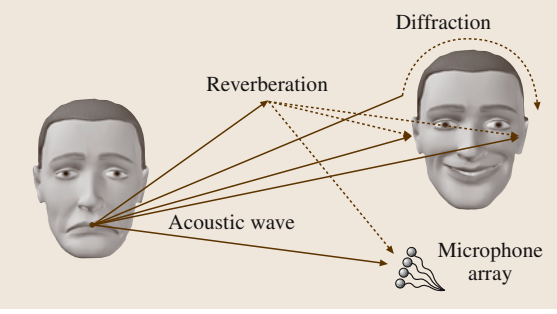


Fig. 52.3 Illustration of a speech source. It is not always clear what the desired acoustic source should be. It could be the acoustic wave as emitted from the mouth. This corresponds to the signal as it would have been recorded in an anechoic chamber in the absence of reverberations. It could be the individual source as it is picked up by a microphone array, or it could be the speech signal as it is recorded on microphones close to the two eardrums of a person. Due to reverberations and diffraction, the recorded speech signal is most likely a filtered version of the signal at the mouth

as illustrated in Fig. 52.2. The criterion for separation, i. e., interference-free signals, is satisfied if the recovered signals are permuted, and possibly scaled and filtered versions of the original signals, i. e.,

$$\boldsymbol{G}(z)=\mathbf{P}\boldsymbol{\Lambda}(z)\,, \tag{52.19}$$

where $\mathbf{P}$ is a permutation matrix, and $\boldsymbol{\Lambda}(z)$ is a diagonal matrix with scaling filters on its diagonal. If one can identify $\boldsymbol{A}(z)$ exactly, and choose $\boldsymbol{W}(z)$ to be its (stable) inverse, then $\boldsymbol{\Lambda}(z)$ is an identity matrix, and one recovers the sources exactly. In source separation, instead, one is satisfied with convolved versions of the sources, i. e., arbitrary diagonal $\boldsymbol{\Lambda}(z)$.

52.2.3 Feedback Structure

The mixing system given by (52.2) is called a feedforward system. Often such FIR filters are inverted by a feedback structure using IIR filters. The estimated sources are then given by the following equation, where the number of sources equals the number of receivers:

$$y_n(t)=x_n(t)+\sum_{l=0}^{L-1}\sum_{m=1}^{M} u_{nml}y_m(t-l)\,, \tag{52.20}$$

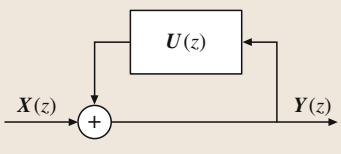


Fig. 52.4 Recurrent unmixing (feedback) network given by equation (52.21). The received signals are separated by an IIR filter to achieve an estimate of the source signals

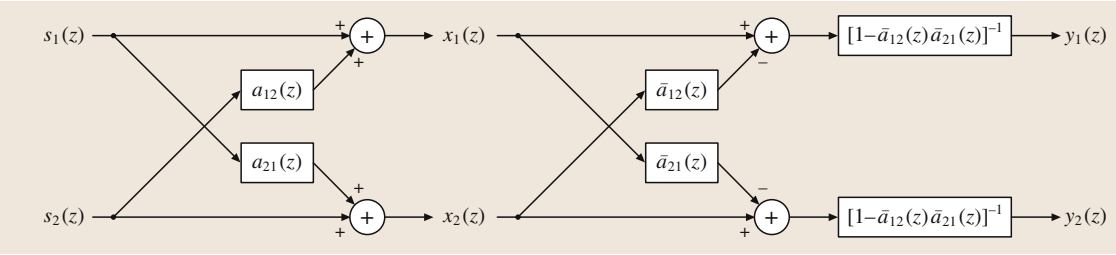


Fig. 52.5 The two mixed sources s_1 and s_2 are mixed by an FIR mixing system. The system can be inverted by an alternative system if the estimates $\bar{a}_{12}(z)$ and $\bar{a}_{21}(z)$ of the mixing filters $a_{12}(z)$ and $a_{12}(z)$ are known. Furthermore, if the filter $[1-\bar{a}_{12}(z)\bar{a}_{21}(z)]^{-1}$ is stable, the sources can be perfectly reconstructed as they were recorded at the microphones

and u_{nml} are the IIR filter coefficients. This can also be written in matrix form

$$\boldsymbol{y}(t) = \boldsymbol{x}(t) + \sum_{l=0}^{L-1} \boldsymbol{U}(l)\boldsymbol{y}(t-l)\,. \tag{52.21}$$

The architecture of such a network is shown in Fig. 52.4. In the z-domain, (52.21) can be written as [52.26]

$$\boldsymbol{Y}(z) = [\boldsymbol{I} + \boldsymbol{U}(z)]^{-1}\boldsymbol{X}(z)\,, \tag{52.22}$$

provided $[\boldsymbol{I} + \boldsymbol{U}(z)]^{-1}$ exists and all poles are within the unit circle. Therefore,

$$\boldsymbol{W}(z) = [\boldsymbol{I} + \boldsymbol{U}(z)]^{-1}\,. \tag{52.23}$$

The feedforward and the feedback network can be combined to a so-called hybrid network, where a feedforward structure is followed by a feedback network [52.27,28].

52.2.4 Example: The TITO System

A special case, which is often used in source separation work is the two-input-two-output (TITO) system [52.29]. It can be used to illustrate the relationship between the mixing and unmixing system, feedforward and feedback structures, and the difference between recovering sources versus generating separated signals.

Figure 52.5 shows a diagram of a TITO mixing and unmixing system. The signals recorded at the two microphones are described by the following equations:

$$x_1(z) = s_1(z) + a_{12}(z)s_2(z)\,, \tag{52.24}$$

$$x_2(z) = s_2(z) + a_{21}(z)s_1(z)\,. \tag{52.25}$$

The mixing system is thus given by

$$\boldsymbol{A}(z) = \begin{pmatrix} 1 & a_{12}(z) \\ a_{21}(z) & 1 \end{pmatrix}\,, \tag{52.26}$$

which has the following inverse

$$[\boldsymbol{A}(z)]^{-1} = \frac{1}{1 - a_{12}(z)a_{21}(z)} \begin{pmatrix} 1 & -a_{12}(z) \\ -a_{21}(z) & 1 \end{pmatrix}\,. \tag{52.27}$$

If the two mixing filters $a_{12}(z)$ and $a_{21}(z)$ can be identified or estimated as $\bar{a}_{12}(z)$ and $\bar{a}_{21}(z)$, the separation system can be implemented as

$$y_1(z) = x_1(z) - \bar{a}_{12}(z)x_2(z) \tag{52.28}$$

$$y_2(z) = x_2(z) - \bar{a}_{21}(z)x_1(z)\,. \tag{52.29}$$

A sufficient FIR separating filter is

$$\boldsymbol{W}(z) = \begin{pmatrix} 1 & -a_{12}(z) \\ -a_{21}(z) & 1 \end{pmatrix}\,. \tag{52.30}$$

However, the exact sources are not recovered until the model sources $\boldsymbol{y}(t)$ are filtered with the IIR filter $[1-\bar{a}_{12}(z)\bar{a}_{21}(z)]^{-1}$. Thus, the mixing process is invertible, provided that the inverse IIR filter is stable. If a filtered version of the separated signals is acceptable, we may disregard the potentially unstable recursive filter in (52.27) and limit separation to the FIR inversion of the mixing system with (52.30).

52.3 Identification

Blind identification deals with the problem of estimating the coefficients in the mixing process $\boldsymbol{A}_k$. In general, this is an ill-posed problem, and no unique solution exists. In order to determine the conditions under which the system is blindly identifiable, assumptions about the mixing process and the input data are necessary. Even though the mixing parameters are known, this does not imply that the sources can be recovered. Blind identification of the sources refers to the exact recovery of sources. Therefore one should distinguish between the conditions required to identify the mixing system and the conditions necessary to identify the sources. The limitations for the exact recovery of sources when the mixing filters are known are discussed in [52.11, 30, 31]. For a recent review on identification of acoustic systems see [52.32]. This review considers systems with single and multiple inputs and outputs for the case of completely known sources as well as blind identification, where both the sources and the mixing channels are unknown.

52.4 Separation Principle

Blind source separation algorithms are based on different assumptions on the sources and the mixing system. In general, the sources are assumed to be *independent* or at least uncorrelated. The separation criteria can be divided into methods based on higher-order statistics (HOS), and methods based on second-order statistics (SOS). In convolutive separation it is also assumed that sensors receive N linearly independent versions of the sources. This means that the sources should originate from different locations in space (or at least emit signals into different orientations) and that there are at least as many sources as sensors for separation, i. e., $M \geq N$.

Instead of spatial diversity a series of algorithms make strong assumptions on the statistics of the sources. For instance, they may require that sources do not overlap in the time–frequency domain, utilizing therefore a form of *sparseness* in the data. Similarly, some algorithms for acoustic mixtures exploit regularity in the sources such as common onset, harmonic structure, etc. These methods are motivated by the present understanding on the grouping principles of auditory perception commonly referred to as *auditory scene analysis*. In radio communications a reasonable assumption on the sources is cyclo-stationarity or the fact that source signals take on only discrete values. By using such strong assumptions on the source statistics it is sometimes possible to relax the conditions on the number of sensors, e.g., $M < N$. The different criteria for separation are summarized in Table 52.2.

Table 52.2 Assumptions made for separation

$N < M$	$N = M$	$N > M$
• Subspace methods [52.25]	• Asymmetric sources by second- and third-order cumulants [52.33]	• Nonstationary, column-wise co-prime sources [52.34]
• Reduction of problem to instantaneous mixture [52.25, 38–43]	• Separation criteria based on SOS and HOS for 2×2 system [52.35]	• Cross-cumulants [52.36, 37]
	• Uncorrelated sources with distinct power spectra [52.44]	• Sparseness in time and frequency [52.45–47]
	• 2×2, temporally colored sources [52.48]	
	• Cumulants of order > 2, ML principle [52.49]	
	• Known cross filters [52.35]	
	• 2×2, each with different correlation [52.50, 51], extended to $M \times M$ in [52.52]	
	• Nonlinear odd functions [52.26, 53–58]	
	• Nonlinearity approximating the cumulative distribution function (CDF), see [52.59]	

52.4.1 Higher-Order Statistics

Source separation based on higher-order statistics is based on the assumption that the sources are statistically independent. Many algorithms are based on minimizing second and fourth order dependence between the model signals. A way to express independence is that all the cross-moments between the model sources are zero, i. e.,

$$E\left[y_n(t)^\alpha,\, y_{n'}(t-\tau)^\beta\right] = 0\,, \tag{52.31}$$

for all τ, $\alpha, \beta = \{1, 2, \dots\}$, and $n \neq n'$. Here $E[\cdot]$ denotes the statistical expectation. Successful separation using higher-order moments requires that the underlying sources are non-Gaussian (with the exception of at most one), since Gaussian sources have zero higher cumulants [52.60] and therefore equations (52.31) are trivially satisfied without providing useful conditions.

Fourth-Order Statistics

It is not necessary to minimize all cross-moments in order to achieve separation. Many algorithms are based on minimization of second- and fourth-order dependence between the model source signals. This minimization can either be based on second and fourth order cross-moments or second- and fourth-order cross-cumulants. Whereas off-diagonal elements of cross-cumulants vanish for independent signals the same is not true for all cross-moments [52.61]. Source separation based on cumulants has been used by several authors. Separation of convolutive mixtures by means of fourth-order cumulants has also been addressed [52.35, 61–71]. In [52.72–74], the joint approximate diagonalization of eigenmatrices (JADE) algorithm for complex-valued signals [52.75] was applied in the frequency domain in order to separate convolved source signals. Other cumulant-based algorithms in the frequency domain are given in [52.76, 77]. Second- and third-order cumulants have been used by Ye et al. [52.33] for separation of asymmetric signals. Other algorithms based on higher-order cumulants can be found in [52.78, 79]. For separation of more sources than sensors, cumulant-based approaches have been proposed in [52.70, 80]. Another popular fourth-order measure of non-Gaussianity is *kurtosis*. Separation of convolutive sources based on kurtosis has been addressed in [52.81–83].

Nonlinear Cross-Moments

Some algorithms apply higher-order statistics for separation of convolutive sources indirectly using nonlinear functions by requiring:

$$E[f(y_n(t)),\, g(y_{n'}(t-\tau))] = 0\,, \tag{52.32}$$

where $f(\cdot)$ and $g(\cdot)$ are odd, nonlinear functions. The Taylor expansion of these functions captures higher-order moments and this is found to be sufficient for separation of convolutive mixtures. This approach was among of the first for separation of convolutive mixtures [52.53] extending an instantaneous blind separation algorithm by *Herault* and *Jutten* (H–J) [52.84]. In *Back* and *Tsoi* [52.85], the H–J algorithm was applied in the frequency domain, and this approach was further developed in [52.86]. In the time domain, the approach of using nonlinear odd functions has been used by *Nguyen Thi* and *Jutten* [52.26]. They present a group of TITO (2×2) algorithms based on fourth-order cumulants, nonlinear odd functions, and second- and fourth-order cross-moments. This algorithm has been further examined by *Serviere* [52.54], and also been used by *Ypma* et al. [52.55]. In *Cruces* and *Castedo* [52.87] a separation algorithm can be found, which can be regarded as a generalization of previous results from [52.26, 88]. In *Li* and *Sejnowski* [52.89], the H–J algorithm has been used to determine the delays in a beamformer. The H–J algorithm has been investigated further by *Charkani* and *Deville* [52.57, 58, 90]. They extended the algorithm further to colored sources [52.56, 91]. Depending on the distribution of the source signals, optimal choices of nonlinear functions were also found. For these algorithms, the mixing process is assumed to be minimum-phase, since the H–J algorithm is implemented as a feedback network. A natural gradient algorithm based on the H–J network has been applied in *Choi* et al. [52.92]. A discussion of the H–J algorithm for convolutive mixtures can be found in *Berthommier* and *Choi* [52.93]. For separation of two speech signals with two microphones, the H–J model fails if the two speakers are located on the same side, as the appropriate separating filters can not be implemented without delaying one of the sources and the FIR filters are constrained to be causal. HOS independence obtained by applying antisymmetric nonlinear functions has also been used in [52.94, 95].

Information-Theoretic Methods

Statistical independence between the source signals can also be expressed in terms of the joint probability function (PDF). If the model sources $\boldsymbol{y}$ are independent, the joint PDF can be written as

$$p(\boldsymbol{y}) = \prod_n p(y_n)\,. \tag{52.33}$$

This is equivalent to stating that model sources y_n do not carry mutual information. Information-theoretic methods for source separation are based on maximizing the entropy in each variable. Maximum entropy is obtained when the sum of the entropy of each variable y_n equals the total joint entropy in $\boldsymbol{y}$. In this limit variables do not carry any mutual information and are hence mutually independent [52.96]. A well-known algorithm based on this idea is the Infomax algorithm by *Bell* and *Sejnowski* [52.97], which was significantly improved in convergence speed by the natural gradient method of *Amari* [52.98]. The Infomax algorithm can also be derived directly from model equation (52.33) using maximum likelihood [52.99], or equivalently, using the Kullback–Leibler divergence between the empirical distribution and the independence model [52.100].

In all instances it is necessary to assume, or model, the probability density functions $p_s(s_n)$ of the underlying sources s_n. In doing so, one captures higher-order statistics of the data. In fact, most information-theoretic algorithms contain expressions rather similar to the nonlinear cross-statistics in (52.32) with $f(y_n) = \partial \ln p_s(y_n)/\partial y_n$, and $g(y_n) = y_n$. The PDF is either assumed to have a specific form or it is estimated directly from the recorded data, leading to *parametric* and *nonparametric* methods, respectively [52.16]. In nonparametric methods the PDF is captured implicitly through the available data. Such methods have been addressed [52.101–103]. However, the vast majority of convolutive algorithms have been derived based on explicit parametric representations of the PDF.

Infomax, the most common parametric method, was extended to the case of convolutive mixtures by *Torkkola* [52.59] and later by *Xi* and *Reilly* [52.104, 105]. Both feedforward and feedback networks were used. In the frequency domain it is necessary to define the PDF for complex variables. The resulting analytic nonlinear functions can be derived as [52.106, 107]

$$f(Y) = -\frac{\partial \ln p(|Y|)}{\partial |Y|} e^{j \arg(Y)} , \qquad (52.34)$$

where $p(Y)$ is the probability density of the model source $Y \in \mathbb{C}$. Some algorithms assume circular sources in the complex domain, while other algorithms have been proposed that specifically assume noncircular sources [52.108, 109].

The performance of the algorithm depends to a certain degree on the selected PDF. It is important to determine if the data has super-Gaussian or sub-Gaussian distributions. For speech commonly a Laplace distribution is used. The nonlinearity is also known as the Bussgang nonlinearity [52.110]. A connection between the Bussgang blind equalization algorithms and the Infomax algorithm is given in *Lambert* and *Bell* [52.111]. Multichannel blind deconvolution algorithms derived from the Bussgang approach can be found in [52.23, 111, 112]. These learning rules are similar to those derived in *Lee* et al. [52.113].

Choi et al. [52.114] have proposed a *nonholonomic* constraint for multichannel blind deconvolution. Nonholonomic means that there are some restrictions related to the direction of the update. The nonholonomic constraint has been applied for both a feedforward and a feedback network. The nonholonomic constraint was applied to allow the natural gradient algorithm by *Amari* et al. [52.98] to cope with overdetermined mixtures. The nonholonomic constraint has also been used in [52.115–122]. Some drawbacks in terms of stability and convergence in particular when there are large power fluctuations within each signal (e.g., for speech) have been addressed in [52.115].

Many algorithms have been derived from (52.33) directly using maximum likelihood (ML) [52.123]. The ML approach has been applied in [52.99, 124–132]. Closely related to the ML are the maximum a posteriori (MAP) methods. In MAP methods, prior information about the parameters of the model are taken into account. MAP has been used in [52.23, 133–141].

The convolutive blind source separation problem has also been expressed in a Bayesian formulation [52.142]. The advantage of a Bayesian formulation is that one can derive an optimal, possibly nonlinear, estimator of the sources enabling the estimation of more sources than the number of available sensors. The Bayesian framework has also been applied [52.135, 137, 143–145].

A strong prior on the signal can also be realized via hidden Markov models (HMMs). HMMs can incorporate state transition probabilities of different sounds [52.136]. A disadvantage of HMMs is that they require prior training and they carry a high computational cost [52.146]. HMMs have also been used in [52.147, 148].

52.4.2 Second-Order Statistics

In some cases, separation can be based on second-order statistics (SOS) by requiring only uncorrelated sources rather then the stronger condition of independence. Instead of assumptions on higher-order statistics these methods make alternate assumptions such as the nonstationarity of the sources [52.149], or a minimum-phase mixing system [52.50]. By itself, however, second-order

conditions are not sufficient for separation. Sufficient conditions for separation are given in [52.15, 150]. The main advantage of SOS is that they are less sensitive to noise and outliers [52.13], and hence require less data for their estimation [52.34, 50, 150–152]. The resulting algorithms are often also easier to implement and computationally efficient.

Minimum-Phase Mixing

Early work by *Gerven* and *Compernolle* [52.88] had shown that two source signals can be separated by decorrelation if the mixing system is minimum-phase. The FIR coupling filters have to be strictly causal and their inverses stable. The condition for stability is given as $|a_{12}(z)a_{21}(z)| < 1$, where $a_{12}(z)$ and $a_{21}(z)$ are the two coupling filters (Fig. 52.5). These conditions are not met if the mixing process is non-minimum-phase [52.153]. Algorithms based on second-order statistic assuming minimum-phase mixing can be found in [52.41, 42, 50–52, 154–158].

Nonstationarity

The fact that many signals are nonstationary has been successfully used for source separation. Speech signals in particular can be considered non-stationary on time scales beyond 10 ms [52.159, 160]). The temporally varying statistics of nonstationarity sources provides additional information for separation. Changing locations of the sources, on the other hand, generally complicate source separation as the mixing channel changes in time. Separation based on decorrelation of nonstationary signals was proposed by *Weinstein* et al. [52.29], who suggested that minimizing cross-powers estimated during different stationarity times should give sufficient conditions for separation. *Wu* and *Principe* proposed a corresponding joint diagonalization algorithm [52.103, 161] extending an earlier method developed for instantaneous mixtures [52.162]. *Kawamoto* et al. extend an earlier method [52.163] for instantaneous mixtures to the case of convolutive mixtures in the time domain [52.153, 164] and frequency domain [52.165]. This approach has also been employed in [52.166–169] and an adaptive algorithm was suggested by *Aichner* et al. [52.170]. By combining this approach with a constraint based on whiteness, the performance can be further improved [52.171].

Note that not all of these papers have used simultaneous decorrelation, yet, to provide sufficient second-order constraints it is necessary to minimize multiple cross-correlations simultaneously. An effective frequency-domain algorithm for simultaneous diagonalization was proposed by *Parra* and *Spence* [52.149]. Second-order statistics in the frequency domain is captured by the cross-power spectrum,

$$\boldsymbol{R}_{yy}(\omega, t) = E[\boldsymbol{Y}(\omega, t)\boldsymbol{Y}^{\mathrm{H}}(\omega, t)] \tag{52.35}$$

$$= \boldsymbol{W}(\omega)\boldsymbol{R}_{xx}(\omega, t)\boldsymbol{W}^{\mathrm{H}}(\omega)\,, \tag{52.36}$$

where the expectations are estimated around some time t. The goal is to minimize the cross-powers represented by the off-diagonal elements of this matrix, e.g., by minimizing:

$$J = \sum_{t,\omega} \|\boldsymbol{R}_{yy}(\omega, t) - \boldsymbol{\Lambda}_y(\omega, t)\|^2\,, \tag{52.37}$$

where $\boldsymbol{\Lambda}_y(\omega, t)$ is an estimate of the cross-power spectrum of the model sources and is assumed to be diagonal. This cost function simultaneously captures multiple times and multiple frequencies, and has to be minimized with respect to $\boldsymbol{W}(\omega)$ and $\boldsymbol{\Lambda}_y(\omega, t)$ subject to some normalization constraint. If the source signals are nonstationary the cross-powers estimated at different times t differ and provide independent conditions on the filters $\boldsymbol{W}(\omega)$. This algorithm has been successfully used on speech signals [52.172, 173] and investigated further by *Ikram* and *Morgan* [52.174–176] to determine the trade-offs between filter length, estimation accuracy, and stationarity times. Long filters are required to cope with long reverberation times of typical room acoustics, and increasing filter length also reduces problems associated with the circular convolution in (52.36) (see Sect. 52.5.3). However, long filters increase the number of parameters to be estimated and extend the effective window of time required for estimating cross-powers, thereby potentially losing the benefit of the nonstationarity of speech signals. A number of variations of this algorithm have been proposed subsequently, including time-domain implementations [52.177–179], and other methods that incorporate additional assumptions [52.174, 180–187]. A recursive version of the algorithm was given in *Ding* et al. [52.188]. In *Robeldo-Arnuncio* and *Juang* [52.189], a version with noncausal separation filters was suggested. Based on a different way to express (52.36), *Wang* et al. [52.148, 190–192] proposed a slightly different separation criterion that leads to a faster convergence than the original algorithm by *Parra* and *Spence* [52.149].

Other methods that exploit nonstationarity have been derived by extending the algorithm of *Molgedey* and *Schuster* [52.193] to the convolutive case [52.194, 195] including a common two-step approach of *sphering* and rotation [52.159, 196–199]. (Any matrix, for instance matrix $\mathbf{W}$, can be represented as a concatenation of a rotation with subsequent scaling, which can be used to

remove second-order moments, i. e., sphering, and an additional rotation.)

In *Yin* and *Sommen* [52.160] a source separation algorithm was presented based on nonstationarity and a model of the direct path. The reverberant signal paths are considered as noise. A time-domain decorrelation algorithm based on different cross-correlations at different time lags is given in *Ahmed* et al. [52.200]. In *Yin* and *Sommen* [52.201] the cost function is based on minimization of the power spectral density between the source estimates. The model is simplified by assuming that the acoustic transfer function between the source and closely spaced microphones is similar. The simplified model requires fewer computations. An algorithm based on joint diagonalization is suggested in *Rahbar* and *Reilly* [52.152, 152]. This approach exploits the spectral correlation between the adjacent frequency bins in addition to nonstationarity. Also in [52.202, 203] a diagonalization criterion based on nonstationarity was used.

In *Olsson* and *Hansen* [52.138, 139] the nonstationary assumption has been included in a state-space Kalman filter model.

In *Buchner* et al. [52.204], an algorithm that uses a combination of non-stationarity, non-Gaussianity, and nonwhiteness has been suggested. This has also been applied in [52.205–207]. In the case of more source signals than sensors, an algorithm based on nonstationarity has also been suggested [52.70]. In this approach, it is possible to separate three signals: a mixture of two nonstationary source signals with short-time stationarity and one signal that is long-term stationary. Other algorithms based on the nonstationary assumptions can be found in [52.208–214].

Cyclo-Stationarity

If a signal is assumed to be cyclo-stationary, its cumulative distribution is invariant with respect to time shifts of some period T multiples thereof. Further, a signal is said to be wide-sense cyclo-stationary if the signals mean and autocorrelation is invariant to shifts of some period T [52.215], i. e.,

$$E[\boldsymbol{s}(t)] = E[\boldsymbol{s}(t+\alpha T)]\,, \tag{52.38}$$

$$E[\boldsymbol{s}(t_1), \boldsymbol{s}(t_2)] = E[\boldsymbol{s}(t_1+\alpha T), \boldsymbol{s}(t_2+\alpha T)]\,. \tag{52.39}$$

An example of a cyclo-stationary signal is a random-amplitude sinusoidal signal. Many communication signals have the property of cyclo-stationarity, and voiced speech is sometimes considered approximately cyclo-stationary [52.216]. This property has been used explicitly to recover mixed sources in, e.g., [52.34, 55, 118, 216–222]. In [52.220] cyclo-stationarity is used to solve the frequency permutation problem (see Sect. 52.5.1) and in [52.118] it is used as additional criteria to improve separation performance.

Nonwhiteness

Many natural signals, in particular acoustic signals, are temporally correlated. Capturing this property can be beneficial for separation. For instance, capturing temporal correlations of the signals can be used to reduce a convolutive problem to an instantaneous mixture problem, which is then solved using additional properties of the signal [52.25, 38–43]. In contrast to instantaneous separation where decorrelation may suffice for nonwhite signals, for convolutive separation additional conditions on the system or the sources are required. For instance, *Mei* and *Yin* [52.223] suggest that decorrelation is sufficient provided the sources are an ARMA process.

52.4.3 Sparseness in the Time/Frequency Domain

Numerous source separation applications are limited by the number of available microphones. It is in not always guaranteed that the number of sources is less than or equal to the number of sensors. With linear filters it is in general not possible to remove more than $M-1$ interfering sources from the signal. By using nonlinear techniques, in contrast, it may be possible to extract a larger number of source signals. One technique to separate more sources than sensors is based on sparseness. If the source signals do not overlap in the time–frequency (T–F) domain it is possible to separate them. A mask can be applied in the T–F domain to attenuate interfering signal energy while preserving T–F bins where the signal of interest is dominant. Often a binary mask is used giving perceptually satisfactory results even for partially overlapping sources [52.224, 225]. These methods work well for anechoic (delay-only) mixtures [52.226]. However, under reverberant conditions, the T–F representation of the signals is less sparse. In a mildly reverberant environment ($T_{60} \leq 200$ ms) underdetermined sources have been separated with a combination of independent component analysis (ICA) and T–F masking [52.47]. The first $N-M$ signals are removed from the mixtures by applying a T–F mask estimated from the direction of arrival of the signal (Sect. 52.6.1). The remaining M sources are separated by conventional BSS techniques. When a binary mask is applied to a signal, artifacts (musical noise) are often introduced. In order to reduce the musical noise, smooth masks have been proposed [52.47, 227].

Sparseness has also been used as a postprocessing step. In [52.77], a binary mask has been applied as post-processing to a standard BSS algorithm. The mask is determined by comparison of the magnitude of the outputs of the BSS algorithm. Hereby a higher signal-to-interference ratio is obtained. This method was further developed by *Pedersen* et al. in order to segregate underdetermined mixtures [52.228, 229]. Because the T–F mask can be applied to a single microphone signal, the segregated signals can be maintained as, e.g., in stereo signals.

Most T–F masking methods do not effectively utilize information from more than two microphones because the T–F masks are applied to a single microphone signal. However, some methods have been proposed that utilize information from more than two microphones [52.225, 230].

Clustering has also been used for sparse source separation [52.140, 141, 230–236]. If the sources are projected into a space where each source groups together, the source separation problem can be solved with clustering algorithms. In [52.45, 46] the mask is determined by clustering with respect to amplitude and delay differences.

In particular when extracting sources from single channels sparseness becomes an essential criterion. *Pearlmutter* and *Zador* [52.237] use strong prior information on the source statistic in addition to knowledge of the head-related transfer functions (HRTF). An a priori dictionary of the source signals as perceived through a HRTF makes it possible to separate source signals with only a single microphone. In [52.238], a priori knowledge is used to construct basis functions for each source signals to segregate different musical signals from their mixture. Similarly, in [52.239, 240] sparseness has been assumed in order to extract different music instruments.

Techniques based on sparseness are further discussed in the survey by *O'Grady* et al. [52.21].

52.4.4 Priors from Auditory Scene Analysis and Psychoacoustics

Some methods rely on insights gained from studies of the auditory system. The work by *Bergman* [52.241] on auditory scene analysis characterized the cues used by humans to segregate sound sources. This motivated computational algorithms that are referred to as computational auditory scene analysis (CASA). For instance, the phenomenon of auditory masking (the dominant perception of the signal with largest power) has motivated the use of T–F masking for many year [52.242]. In addition to the direct T–F masking methods outlined above, separated sources have been enhanced by filtering based on perceptual masking and auditory hearing thresholds [52.191, 243].

Another important perceptual cue that has been used in source separation is pitch frequency, which typically differs for simultaneous speakers [52.135, 137, 138, 147, 244, 245]. In *Tordini* and *Piazza* [52.135] pitch is extracted from the signals and used in a Bayesian framework. During unvoiced speech, which lacks a well-defined pitch they use an ordinary blind algorithm. In order to separate two signals with one microphone, *Gandhi* and *Hasegawa-Johnson* [52.137] proposed a state-space separation approach with strong *a priori* information. Both pitch and mel-frequency cepstral coefficients (MFCC) were used in their method. A pitch codebook as well as an MFCC codebook have to be known in advance. *Olsson* and *Hansen* [52.138] used an HMM, where the sequence of possible states is limited by the pitch frequency that is extracted in the process. As a preprocessing step to source separation, *Furukawa* et al. [52.245] use pitch in order to determine the number of source signals.

A method for separation of more sources than sensors is given in *Barros* et al. [52.244]. They combined ICA with CASA techniques such as pitch tracking and auditory filtering. Auditory filter banks are used in order to model the cochlea. In [52.244] wavelet filtering has been used for auditory filtering. Another commonly used auditory filter bank is the Gammatone filter-bank (see, e.g., *Patterson* [52.246] or [52.247, 248]). In *Roman* et al. [52.248] binaural cues have been used to segregate sound sources, whereby interaural time and interaural intensity differences (ITD and IID) have been used to group the source signals.

52.5 Time Versus Frequency Domain

The blind source separation problem can either be expressed in the time domain

$$\boldsymbol{y}(t) = \sum_{l=0}^{L-1} \boldsymbol{W}_l \boldsymbol{x}(t-l) \tag{52.40}$$

or in the frequency domain

$$\boldsymbol{Y}(\omega, t) = \boldsymbol{W}(\omega)\boldsymbol{X}(\omega, t)\,. \tag{52.41}$$

A survey of frequency-domain BSS is provided in [52.22]. In *Nishikawa* et al. [52.249] the advantages and disadvantages of the time and frequency-domain approaches are compared. This is summarized in Table 52.3.

An advantage of blind source separation in the frequency domain is that the separation problem can be decomposed into smaller problems for each frequency bin in addition to the significant gains in computational efficiency. The convolutive mixture problem is reduced to *instantaneous* mixtures for each frequency. Although this simplifies the task of convolutive separation a set of new problems arise: the frequency-domain signals obtained from the DFT are complex-valued. Not all instantaneous separation algorithms are designed for complex-valued signals. Consequently, it is necessary to modify existing algorithms correspondingly [52.5, 250–252]. Another problem that may arise in the frequency domain is that there are no longer enough data points available to evaluate statistical independence [52.131]. For some algorithms [52.149] the frame size T of the DFT has to be much longer than the length of the room impulse response K (Sect. 52.5.3). Long frames result in fewer data samples per frequency [52.131], which complicates the estimation of the independence criteria. A method that addresses this issue has been proposed by *Servière* [52.253].

52.5.1 Frequency Permutations

Another problem that arises in the frequency domain is the permutation and scaling ambiguity. If separation is treated for each frequency bin as a separate problem, the source signals in each bin may be estimated with an arbitrary permutation and scaling, i. e.,

$$\boldsymbol{Y}(\omega, t) = \boldsymbol{P}(\omega)\boldsymbol{\Lambda}(\omega)\boldsymbol{S}(\omega, t)\,. \tag{52.42}$$

If the permutation $\boldsymbol{P}(\omega)$ is not consistent across frequency then converting the signal back to the time domain will combine contributions from different sources into a single channel, and thus annihilate the separation achieved in the frequency domain. An overview of the solutions to this permutation problem is given in Sect. 52.6. The scaling indeterminacy at each frequency – the arbitrary solution for $\boldsymbol{\Lambda}(\omega)$ – will result in an overall filtering of the sources. Hence, even for perfect separation, the separated sources may have a different frequency spectrum than the original sources.

52.5.2 Time–Frequency Algorithms

Algorithms that define a separation criteria in the time domain do typically not exhibit frequency permutation problems, even when computations are executed in the frequency domain. A number of authors have therefore used time-domain (TD) criteria combined with frequency-domain implementations that speed up computations [52.101, 113, 121, 171, 179, 254–257]. However, note that second-order criteria may be sus-

Table 52.3 Advantages and disadvantages of separation in the time and frequency domain

Time domain		**Frequency domain**	
Advantages	Disadvantages	Advantages	Disadvantages
The independence assumption holds better for full-band signals	Degradation of convergence in a strongly reverberant environment	The convolutive mixture can be transformed into instantaneous mixture problems for each frequency bin	For each frequency band, there is a permutation and a scaling ambiguity which needs to be solved
Possible high convergence near the optimal point	Many parameters need to be adjusted for each iteration step	Due to the FFT, computations are saved compared to an implementation in the time domain	Problem with too few samples in each frequency band may cause the independence assumption to fail
		Convergence is faster	Circular convolution deteriorates the separation performance.
			Inversion of $\boldsymbol{A}$ is not guaranteed

ceptible to the permutation problem even if they are formulated in the time domain [52.184].

52.5.3 Circularity Problem

When the convolutive mixture in the time domain is expressed in the frequency domain by the DFT, the convolution becomes separate multiplications, i. e.,

$$\boldsymbol{x}(t) = \boldsymbol{A} * \boldsymbol{s}(t) \leftrightarrow \boldsymbol{X}(\omega, t) \approx \boldsymbol{A}(\omega)\boldsymbol{S}(\omega, t)\,. \quad (52.43)$$

However, this is only an approximation which is exact only for periodic $\boldsymbol{s}(t)$ with period T, or equivalently, if the time convolution is *circular*:

$$\boldsymbol{x}(t) = \boldsymbol{A} \circledast \boldsymbol{s}(t) \longleftrightarrow \boldsymbol{X}(\omega) = \boldsymbol{A}(\omega)\boldsymbol{S}(\omega)\,. \quad (52.44)$$

For a *linear convolution* errors occur at the frame boundary, which are conventionally corrected with the overlap-save method. However, a correct overlap-save algorithm is difficult to implement when computing cross-powers such as in (52.36) and typically the approximate expression (52.43) is assumed.

The problem of linear/circular convolution has been addressed by several authors [52.62, 121, 149, 171, 258]. *Parra* and *Spence* [52.149] note that the frequency-domain approximation is satisfactory provided that the DFT length T is significantly larger than the length of the un-mixing channels. In order to reduce the errors due to the circular convolution, the filters should be at least two times the length of the un-mixing filters [52.131, 176].

To handle long impulse responses in the frequency domain, a frequency model which is equivalent to the time-domain linear convolution has been proposed in [52.253]. When the time-domain filter extends beyond the analysis window the frequency response is undersampled [52.22, 258]. These errors can be mitigated by spectral smoothing or equivalently by windowing in the time domain. According to [52.259] the circularity problem becomes more severe when the number of sources increases.

Time-domain algorithms are often derived using Toeplitz matrices. In order to decrease the complexity and improve computational speed, some calculations involving Toeplitz matrices are performed using the fast Fourier transform. For that purpose, it is necessary to express the Toeplitz matrices in circulant Toeplitz form [52.23, 121, 171, 195, 260, 261]. A method that avoids the circularity effects but maintains the computational efficiency of the FFT has been presented in [52.262]. Further discussion on the circularity problem can be found in [52.189].

52.5.4 Subband Filtering

Instead of the conventional linear Fourier domain some authors have used subband processing. In [52.142] a long time-domain filter is replaced by a set of short independent subband filters, which results in faster convergence as compared to the full-band methods [52.214]. Different filter lengths for each subband filter have also been proposed, motivated by the varying reverberation time of different frequencies (typically low frequencies have a longer reverberation time) [52.263].

52.6 The Permutation Ambiguity

The majority of algorithms operate in the frequency domain due to the gains in computational efficiency, which are important in particular for acoustic mixtures that require long filters. However, in frequency-domain algorithms the challenge is to solve the permutation ambiguity, i. e., to make the permutation matrix $\boldsymbol{P}(\omega)$ independent of frequency. Especially when the number of sources and sensors is large, recovering consistent permutations is a severe problem. With N model sources there are $N!$ possible permutations in each frequency bin. Many frequency-domain algorithms provide ad hoc solutions, which solve the permutation ambiguity only partially, thus requiring a combination of different methods. Table 52.4 summarizes different approaches. They can be grouped into two categories:

1. Consistency of the filter coefficients
2. Consistency of the spectrum of the recovered signals

The first exploits prior knowledge about the mixing filters, and the second uses prior knowledge about the sources. Within each group the methods differ in the way consistency across frequency is established, varying sometimes in the metric they use to measure *distance* between solutions at different frequencies.

52.6.1 Consistency of the Filter Coefficients

Different methods have been used to establish consistency of filter coefficients across frequency, such as

Table 52.4 Categorization of approaches to solve the permutation problem in the frequency domain

Class	Metric	Reference
Consistency of the filter coefficients	Smooth spectrum	[52.149, 264]
	Source locations	[52.265]
	Directivity pattern	[52.73, 175, 266]
	Location vectors	[52.267]
	DOA	[52.72, 184, 268]
	Adjacent matrix distance	[52.269]
	Invariances	[52.48]
	Split spectrum	[52.270]
	Frequency link in update process	[52.127]
	Initialization	[52.250, 271]
	Moving sources	[52.167]
	Vision	[52.148]
Consistency of the spectrum of the recovered signals	Amplitude modulation	[52.126, 159, 197, 203, 272]
	Pitch	[52.135, 147]
	Psychoacoustics	[52.243, 243]
	Fundamental frequency	[52.244]
	Cyclo-stationarity	[52.273]
	Periodic signals	[52.221]
	Cross-correlation	[52.62, 209, 274]
	Cross-cumulants	[52.275]
	Kurtosis	[52.86]
	Source distribution	[52.134, 276]
	Multidimensional prior	[52.277, 278]
	Clustering	[52.230, 279]
Time-frequency information	FIR polynomial	[52.23, 113, 254, 255]
	TD cost function	[52.178]
	Apply ICA to whole spectrogram	[52.280]
Combined approaches		[52.106, 258, 281, 282]

constraints on the length of the filters, geometric information, or consistent initialization of the filter weights.

Consistency across frequency can be achieved by requiring continuity of filter values in the frequency domain. One may do this directly by comparing the filter values of neighboring frequencies after adaptation, and pick the permutation that minimize the Euclidean distance between neighboring frequencies [52.74, 269]. Continuity (in a discrete frequency domain) is also expressed as smoothness, which is equivalent with a limited temporal support of the filters in the time domain. The simplest way to implement such a smoothness constraint is by zero-padding the time-domain filters prior to performing the frequency transformation [52.264]. Equivalently, one can restrict the frequency-domain updates to have a limited support in the time domain. This method is explained in *Parra* et al. [52.149] and has been used extensively [52.119, 122, 161, 174, 188, 190, 192, 201, 269, 283]. *Ikram* and *Morgan* [52.174, 176] evaluated this constraint and point out that there is a trade-off between the permutation alignment and the spectral resolution of the filters. Moreover, restricting the filter length may be problematic in reverberant environments where long separation filters are required. As a solution they have suggest to relax the constraint on filter length after the algorithm converges to satisfactory solutions [52.176].

Another suggestion is to assess continuity after accounting for the arbitrary scaling ambiguity. To do so, the separation matrix can be normalized as proposed in [52.265]:

$$\boldsymbol{W}(\omega) = \widetilde{\boldsymbol{W}}(\omega)\boldsymbol{\Lambda}(\omega)\,, \tag{52.45}$$

where $\boldsymbol{\Lambda}(\omega)$ is a diagonal matrix and $\widetilde{\boldsymbol{W}}(\omega)$ is a matrix with unit diagonal. The elements of $\widetilde{\boldsymbol{W}}(\omega)$, $\widetilde{W}_{mn}(\omega)$ are the ratios between the filters and these are used to assess continuity across frequencies [52.48, 220].

Instead of restricting the *unmixing* filters, *Pham* et al. [52.202] have suggested to require continuity in the *mixing* filters, which is reasonable as the mixing process will typically have a shorter time constant. A specific distance measure has been proposed by *Asano* et al. [52.267, 284]. They suggest to use the cosine between the filter coefficients of different frequencies ω_1 and ω_2:

$$\cos\alpha_n = \frac{\boldsymbol{a}_n^{\mathrm{H}}(\omega_1)\boldsymbol{a}_n(\omega_2)}{\left\|\boldsymbol{a}_n^{H}(\omega_1)\right\|\left\|\boldsymbol{a}_n(\omega_2)\right\|}\,, \tag{52.46}$$

where $\boldsymbol{a}_n(\omega)$ is the n-th column vector of $\boldsymbol{A}(\omega)$, which is estimated as the pseudo-inverse of $\boldsymbol{W}(\omega)$. Measuring distance in the space of separation filters rather than mixing filters was also suggested because these may better reflect the spacial configuration of the sources [52.285].

In fact, continuity across frequencies may also be assessed in terms of the estimated spatial locations of sources. Recall that the mixing filters are impulse responses between the source locations and the microphone locations. Therefore, the parameters of the separation filters should account for the position of the source in space. Hence, if information about the sensor location is available it can be used to address the permutation problem.

To understand this, consider the signal that arrives at an array of sensors. Assuming a distant source in an reverberation-free environment the signal approximates a plane wave. If the plane wave arrives at an angle to the microphone array it will impinge on each microphone with a certain delay (Fig. 52.6). This delay τ is given by the microphone distance d, the velocity of the wave c, and the direction-of-arrival (DOA) angle θ:

$$\tau = \frac{d}{c}\sin\theta\,, \tag{52.47}$$

Filters that compensate for this delay can add the microphone signals constructively (or destructively) to produce a maximum (or minimum) response in the DOA. Hence, the precise delay in filters (which in the frequency domain correspond to precise phase relationships) establishes a relationship between different frequencies that can be used to identify correct permutations. This was first considered by *Soon* et al. [52.286].

To be specific, each row in the separation matrix $\mathbf{W}(\omega)$ defines a *directivity pattern*, and therefore each row can be thought of as a separate beamformer. This

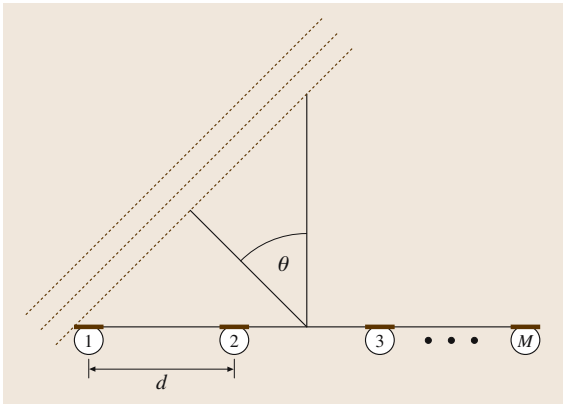


Fig. 52.6 Linear array with M sensors separated by distance d. The sensors are placed in a free field. A source signal is considered coming from a point source a distance r away from the sensor array. The source signal is placed in the far field, i. e., $r \gg d$. Therefore the incident wave is planar and the arrival angle θ is the same for all the sensors

directivity pattern is determined by the transfer function between the source and the filter output. The magnitude response of the n-th output is given by

$$\boldsymbol{r}_n(\omega,\boldsymbol{\theta}) = \left|\boldsymbol{w}_n^{\mathrm{H}}(\omega)\boldsymbol{a}(\omega,\boldsymbol{\theta})\right|^2\,, \tag{52.48}$$

where $\boldsymbol{a}(\omega)$ is an $M\times 1$ vector representing the propagation of a distant source with DOA θ to the sensor array. When M sensors are available, it is possible to place $M-1$ nulls in each of the M *directivity patterns*, i. e., directions from which the arriving signal is canceled out. In an ideal, reverberation-free environment separation is achieved if these nulls point to the directions of the interfering sources. The locations of these nulls, as they may be identified by the separation algorithm, can be used to resolve the permutation ambiguity [52.77, 81, 131, 266, 287–290]. These techniques draw strong parallels between source separation solutions and *beamforming*. The DOAs do not have to be known in advance and can instead be extracted from the resulting separation filters. Note, however, that the ability to identify source locations is limited by the physics of wave propagation and sampling: distant microphones will lead to grading lobes which will confuse the source locations, while small aperture limits spatial resolution at low frequencies. *Ikram* and *Morgan* [52.175] extend the idea of *Kurita* et al. [52.266] to the case where the sensor space is wider than one half of the wavelength. Source locations are estimated at lower frequencies, which do not exhibit grating lobes. These estimates are then used to determine the correct nulls for the higher fre-

quencies and hereby the correct permutations. In order to resolve permutations when sources arrive from the same direction, *Mukai* et al. [52.291] use a near-field model. *Mitianoudis* and *Davies* [52.268] suggested frequency alignment based on DOA estimated with the multiple signal classification (MuSIC) algorithm [52.292]. A subspace method has been used in order to avoid constraints on the number of sensors. *Knaak* et al. [52.222] include DOA information as a part of the BSS algorithm in order to avoid permutation errors. Although all these methods assume a reverberation-free environment they give reasonable results in reverberant environments as long as the source has a strong direct path to the sensors.

Two other methods also utilize geometry. In the case of moving sources, where only one source is moving, the permutation can be resolved by noting that only one of the parameters in the separation matrix changes [52.167]. If visual cues are available, they may also be used to solve the permutation ambiguity [52.148].

Instead of using geometric information as a separate step to solve the permutation problem *Parra* and *Alvino* include geometric information directly into the cost function [52.184, 185]. This approach has been applied to microphone arrays under reverberant conditions [52.187]. *Baumann* et al. [52.72] have also suggested a cost function, which includes the DOA estimation. The arrival angles of the signals are found iteratively and are included in the separation criterion. *Baumann* et al. [52.73] also suggest a maximum-likelihood approach to solve the permutation problem. Given the probability of a permuted or unpermuted solution as function of the estimated zero directions, the most likely permutation is found.

Gotanda et al. [52.270] proposed a method to reduce the permutation problem based on the split spectral difference, and the assumption that each source is closer to one microphone. The split spectrum is obtained when each of the separated signals are filtered by the estimated mixing channels.

Finally, for iterative update algorithms a proper initialization of the separation filters can result in consistent permutations across frequencies. *Smaragdis* [52.250] proposed to estimate filter values sequentially starting with low frequencies and initializing filter values with the results of the previous lower frequency. This will tend to select solutions with filters that are smooth in the frequency domain, or equivalently, filters that are short in the time domain. Filter values may also be initialized to simple beamforming filters that point to estimated source locations. The separation algorithm will then tend to converge to solutions with the same target source across all frequencies [52.184, 271].

52.6.2 Consistency of the Spectrum of the Recovered Signals

Some solutions to the permutation ambiguity are based on the properties of speech. Speech signals have strong correlations across frequency due to a common amplitude modulation.

At the coarsest level the power envelope of the speech signal changes depending on whether there is speech or silence, and within speech segments the power of the carrier signal induces correlations among the amplitude of different frequencies. A similar argument can be made for other natural sounds. Thus, it is fair to assume that natural acoustic signals originating from the same source have a correlated amplitude envelope for neighboring frequencies. A method based on this comodulation property was proposed by *Murata* et al. [52.159, 196]. The permutations are sorted to maximize the correlation between different envelopes. This is illustrated in Fig. 52.7. This method has also been used in [52.198, 199, 203, 263, 287, 293]. *Rahbar* and *Reilly* [52.152, 209] suggest efficient methods for finding the correct permutations based on cross-frequency correlations.

Asano and *Ikeda* [52.294] report that the method sometimes fails if the envelopes of the different source signals are similar. They propose the following function to be maximized in order to estimate the permutation

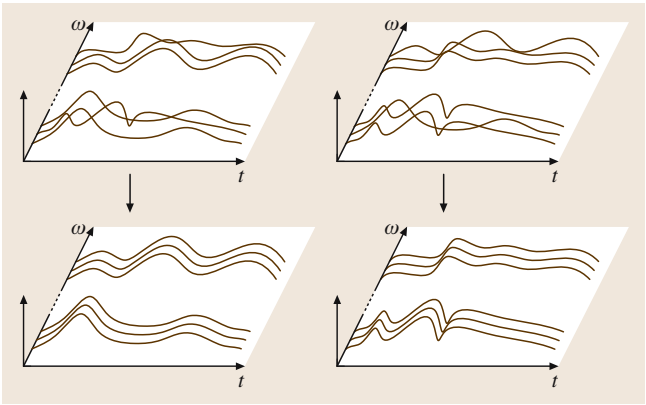


Fig. 52.7 For speech signals, it is possible to estimate the permutation matrix by using information on the envelope of the speech signal (amplitude modulation). Each speech signal has a particular envelope. Therefore, by comparison with the envelopes of the nearby frequencies, it is possible to order the permuted signals

matrix:

$$\hat{\boldsymbol{P}}(\omega) = \arg\max_{\boldsymbol{P}(\omega)} \sum_{t=1}^{T} \sum_{j=1}^{\omega-1} [\boldsymbol{P}(\omega)\bar{\boldsymbol{y}}(\omega, t)]^{\mathrm{H}} \bar{\boldsymbol{y}}(j, t) \,, \tag{52.49}$$

where $\bar{\boldsymbol{y}}$ is the power envelope of $\boldsymbol{y}$ and $\boldsymbol{P}(\omega)$ is the permutation matrix. This approach has also been adopted by *Peterson* and *Kadambe* [52.232]. *Kamata* et al. [52.282] report that the correlation between envelopes of different frequency channels may be small, if the frequencies are too far from each other. *Anemüller* and *Gramms* [52.127] avoid the permutations since the different frequencies are linked in the update process. This is done by serially switching from low to high-frequency components while updating.

Another solution based on amplitude correlation is the so-called amplitude modulation decorrelation (AMDecor) algorithm presented by *Anemüller* and *Kollmeier* [52.126, 272]. They propose to solve the source separation problem and the permutation problems simultaneously. An amplitude modulation correlation is defined, where the correlation between the frequency channels ω_k and ω_l of the two spectrograms $\boldsymbol{Y}_\mathrm{a}(\omega, t)$ and $\boldsymbol{Y}_\mathrm{b}(\omega, t)$ is calculated as

$$\begin{aligned} c(\boldsymbol{Y}_\mathrm{a}(\omega, t), \boldsymbol{Y}_\mathrm{b}(\omega, t)) = {} & E[|\boldsymbol{Y}_\mathrm{a}(\omega, t)||\boldsymbol{Y}_\mathrm{b}(\omega, t)|] \\ & - E[|\boldsymbol{Y}_\mathrm{a}(\omega, t)|]E[|\boldsymbol{Y}_\mathrm{b}(\omega, t)|] \,. \end{aligned} \tag{52.50}$$

This correlation can be computed for all combinations of frequencies. This results in a square matrix $\boldsymbol{C}(\boldsymbol{Y}_\mathrm{a}, \boldsymbol{Y}_\mathrm{b})$ with sizes equal to the number of frequencies in the spectrogram, whose k, l-th element is given by (52.50). Since the unmixed signals $\boldsymbol{y}(t)$ have to be independent, the following decorrelation property must be fulfilled

$$C_{kl}(\boldsymbol{Y}_\mathrm{a}, \boldsymbol{Y}_\mathrm{b}) = 0 \quad \forall a \neq b, \forall k, l. \tag{52.51}$$

This principle also solves the permutation ambiguity. The source separation algorithm is then based on the minimization of a cost function given by the Frobenius norm of the amplitude-modulation correlation matrix.

A priori knowledge about the source distributions has also been used to determine the correct permutations. Based on assumptions of Laplacian distributed sources, *Mitianopudis* and *Davies* [52.134, 251, 276] propose a likelihood ratio test to test which permutation is most likely. A time-dependent function that imposes frequency coupling between frequency bins is also introduced. Based on the same principle, the method has been extended to more than two sources by *Rahbar* and *Reilly* [52.152]. A hierarchical sorting is used in order to avoid errors introduced at a single frequency. This approach has been adopted in *Mertins* and *Russel* [52.212].

Finally, one of the most effective convolutive BSS methods to date (Table 52.5) uses the statistical relationship of signal powers across frequencies. Rather than solving separate *instantaneous* source separation problems in each frequency band *Lee* et al. [52.277,278,280] propose a multidimensional version of the density estimation algorithms. The density function captures the power of the entire model source rather than the power at individual frequencies. As a result, the joint statistics across frequencies are effectively captured and the algorithm converges to satisfactory permutations in each frequency bin.

Other properties of speech have also been suggested in order to solve the permutation indeterminacy. A *pitch*-based method has been suggested by *Tordini* and *Piazza* [52.135]. Also *Sanei* et al. [52.147] use the property of different pitch frequency for each speaker. The pitch and formants are modeled by a coupled HMM. The model is trained based on previous time frames.

Motivated by psychoacoustics, *Guddeti* and *Mulgrew* [52.243] suggest to disregard frequency bands that are perceptually masked by other frequency bands. This simplifies the permutation problem as the number of frequency bins that have to be considered is reduced. In *Barros* et al. [52.244], the permutation ambiguity is avoided due to a priori information of the phase associated with the fundamental frequency of the desired speech signal.

Nonspeech signals typically also have properties which can be exploited. Two proposals for solving the

Table 52.5 An overview of algorithms applied in real rooms, where the SIR improvement has been reported

Room size (approx.) [m]	T_{60} [ms]	N	M	SIR [dB]	Reference
6×3×3	300	2	2	13	[52.169,170][1]
6×3×3	300	2	2	8–10	[52.271][1]
6×3×3	300	2	2	12	[52.249]
6×3×3	300	2	2	5.7	[52.290]
6×3×3	300	2	2	18–20	[52.132,295][1]
	50	2	2	10	[52.207]
	250	2	2	16	[52.262]
6×6×3	200	2	2	< 16	[52.205][2]
6×6×3	150	2	2	< 15	[52.206]
6×6×3	150	2	2	< 20	[52.171]

Table 52.5 (continued)

Room size (approx.) [m]	T_{60} [ms]	N	M	SIR [dB]	Reference
	500	2	2	6	[52.262]
4×4×3	130	3	2	4–12	[52.296]
4×4×3	130	3	2	14.3	[52.227]
4×4×3	130	3	2	< 12	[52.47]
4×4×3	130	2	2	7–15	[52.130]
4×4×3	130	2	2	4–15	[52.22, 297][2]
4×4×3	130	2	2	12	[52.291]
4×4×3	130	6	8	18	[52.298]
4×4×3	130	4	4	12	[52.259]
	130	3	2	10	[52.140, 141]
Office		2	2	5.5–7.6	[52.142]
6×5	130	2	8	1.6–7.0	[52.269]
8×7	300	2	2	4.2–6.0	[52.73]
15×10	300	2	2	5–8.0	[52.72]
		2	2	< 10	[52.57, 91]
Office		2	2	6	[52.122]
Many rooms		2	2	3.1–27.4	[52.115]
Small room		2	2	4.7–9.5	[52.252]
4×3×2		2	2	< 10	[52.181]
4×3×2		2	2	14.4	[52.183]
4×3×2		2	2	4.6	[52.182]
		2	2	< 15	[52.245]
6×7	580	2	3	< 73	[52.31][3]
	810	2	2	< 10	[52.167][2]
Conf. room		4	4	14	[52.278]
	150	3	3	10	[52.222]
15×10×4	300	2	2	10	[52.77]
	360	2	2	5	[52.266]
5×5	200	2	2	6–21	[52.299]
	300	2	2–12	8–12	[52.300]
3×6		3	8	10	[52.184]
4×3×2		2	2	15	[52.149]
5×5×3		2	2	5	[52.187]
8×4	700	2	4	16	[52.152]
7×4×3	250	2	2	9.3	[52.253][1]
4×4	200	2	2	15	[52.301]
Office	500	3	2	4.3–10.6	[52.45]
	300	2	6	< 15	[52.213]

[1] Sources convolved with real impulse responses

[2] Moving sources

[3] This method is not really blind as it requires that sources are on one at a time

permutation in the case of cyclo-stationary signals can be found in *Antoni* et al. [52.273]. For machine acoustics, the permutations can be solved easily since machine signals are (quasi)periodic. This can be employed to find the right component in the output vector [52.221].

Continuity of the frequency spectra has been used by *Capdevielle* et al. [52.62] to solve the permutation ambiguity. The idea is to consider the sliding Fourier transform with a delay of one point. The cross-correlation between different sources are zero due to the independence assumption. Hence, when the cross-correlation is maximized, the output belongs to the same source. This method has also been used by *Servière* [52.253]. A disadvantage of this method is that it is computationally very expensive since the frequency spectrum has to be calculated with a window shift of one. A computationally less expensive method based on this principle has been suggested by *Dapena* and *Servière* [52.274]. The permutation is determined from the solution that maximizes the correlation between only two frequencies. If the sources have been whitened as part of separation, the approach by *Capdevielle* et al. [52.62] does not work. Instead, *Kopriva* et al. [52.86] suggest that the permutation can be solved by independence tests based on kurtosis. For the same reason, *Mejuto* et al. [52.275] consider fourth-order cross-cumulants of the outputs at all frequencies. If the extracted sources belong to the same sources, the cross-cumulants will be nonzero. Otherwise, if the sources belong to different sources, the cross-cumulants will be zero.

Finally, *Hoya* et al. [52.302] use pattern recognition to identify speech pauses that are common across frequencies, and in the case of overcomplete source separation, k-means clustering has been suggested. The clusters with the smallest variance are assumed to correspond to the desired sources [52.230]. *Dubnov* et al. [52.279] also address the case of more sources than sensors. Clustering is used at each frequency and Kalman tracking is performed in order to link the frequencies together.

52.6.3 Global Permutations

In many applications only one of the source signals is desired and the other sources are only considered as interfering noise. Even though the local (frequency) permutations are solved, the global (external) permutation problem still exists. Few algorithms address the problem of selecting the desired source signal from the available outputs. In some situations, it can be assumed that the desired signal arrives from a certain direction (e.g., the

speaker of interest is in front of the array). Geometric information can determine which of the signals is the target [52.171, 184]. In other situations, the desired speaker is selected as the most dominant speaker. In *Low* et al. [52.289], the most dominant speaker is determined on a criterion based on kurtosis. The speaker with the highest kurtosis is assumed to be the dominant. In separation techniques based on clustering, the desired source is assumed to be the cluster with the smallest variance [52.230]. If the sources are moving it is necessary to maintain the global permutation by tracking each source. For block-based algorithm the global permutation might change at block boundaries. This problem can often be solved by initializing the filter with the estimated filter from the previous block [52.186].

52.7 Results

The overwhelming majority of convolutive source separation algorithms have been evaluated on simulated data. In the process, a variety of simulated room responses have been used. Unfortunately, it is not clear whether any of these results transfer to real data. The main concerns are the sensitivity to microphone noise (often not better than −25 dB), nonlinearity in the sensors, and strong reverberations with a possibly weak direct path. It is suggestive that only a small subset of research teams evaluate their algorithms on actual recordings. We have considered more than 400 references and found results on real room recordings in only 10% of the papers. Table 52.5 shows a complete list of those papers. The results are reported as signal-to-interference ratio (SIR), which is typically averaged over multiple output channels. The resulting SIR are not directly comparable as the results for a given algorithm are very likely to depend on the recording equipment, the room that was used, and the SIR in the recorded mixtures. A state-of-the art algorithm can be expected to improve the SIR by 10–20 dB for two stationary sources. Typically a few seconds of data (2–10 s) will be sufficient to generate these results. However, from this survey nothing can be said about moving sources. Note that only eight (of over 400) papers reported separation of more than two sources, indicating that this remains a challenging problem.

52.8 Conclusion

We have presented a taxonomy for blind separation of convolutive mixtures with the purpose of providing a survey and discussion of existing methods. Further, we hope that this might stimulate the development of new models and algorithms which more efficiently incorporate specific domain knowledge and useful prior information.

In the title of the BSS review by *Torkkola* [52.13], it was asked: *Are we there yet?* Since then numerous algorithms have been proposed for blind separation of convolutive mixtures. Many convolutive algorithms have shown good performance when the mixing process is stationary, but still few methods work in real-world, time-varying environments. In these challenging environments, there are too many parameters to update in the separation filters, and too little data available in order to estimate the parameters reliably, while the less complicated methods such as null beamformers may perform just as well. This may indicate that the long demixing filters are not the solution for real-world, time-varying environments such as the cocktail-party situation.

References

52.1 A. Mansour, N. Benchekroun, C. Gervaise: Blind separation of underwater acoustic signals, ICA'06 (2006) pp. 181–188

52.2 S. Cruces-Alvarez, A. Cichocki, L. Castedo-Ribas: An iterative inversion approach to blind source separation, IEEE Trans. Neural Netw. **11**(6), 1423–1437 (2000)

52.3 K.I. Diamantaras, T. Papadimitriou: MIMO blind deconvoluition using subspace-based filter deflation, Proc. ICASSP'04, Vol. IV (2004) pp. 433–436

52.4 D. Nuzillard, A. Bijaoui: Blind source separation and analysis of multispectral astronomical images, Astron. Astrophys. Suppl. Ser. **147**, 129–138 (2000)

52.5 J. Anemüller, T.J. Sejnowski, S. Makeig: Complex independent component analysis of frequency-domain electroencephalographic data, Neural Netw. **16**(9), 1311–1323 (2003)

52.6 M. Dyrholm, S. Makeig, L.K. Hansen: Model structure selection in convolutive mixtures, ICA'06 (2006) pp. 74–81

52.7 C. Vayá, J.J. Rieta, C. Sánchez, D. Moratal: Performance study of convolutive BSS algorithms applied to the electrocardiogram of atrial fibrillation, ICA'06 (2006) pp. 495–502

52.8 L.K. Hansen: ICA of fMRI based on a convolutive mixture model, Ninth Annual Meeting of the Organization for Human Brain Mapping (2003)

52.9 E.C. Cherry: Some experiments on the recognition of speech, with one and two ears, J. Acoust. Soc. Am. **25**(5), 975–979 (1953)

52.10 S. Haykin, Z. Chen: The cocktail party problem, Neural Comput. **17**, 1875–1902 (2005)

52.11 M. Miyoshi, Y. Kaneda: Inverse filtering of room acoustics, IEEE Trans. Acoust. Speech. Signal Proc. **36**(2), 145–152 (1988)

52.12 K.J. Pope, R.E. Bogner: Blind signal separation II. Linear convolutive combinations, Digital Signal Process. **6**, 17–28 (1996)

52.13 K. Torkkola: Blind separation for audio signals – are we there yet?, ICA'99 (1999) pp. 239–244

52.14 K. Torkkola: Blind separation of delayed and convolved sources. In: *Unsupervised Adaptive Filtering, Blind Source Separation*, Vol. 1, ed. by S. Haykin (Wiley, New York 2000) pp. 321–375, Chap. 8

52.15 R. Liu, Y. Inouye, H. Luo: A system-theoretic foundation for blind signal separation of MIMO-FIR convolutive mixtures – a review, ICA'00 (2000) pp. 205–210

52.16 K.E. Hild: Blind Separation of Convolutive Mixtures Using Renyi's Divergence. Ph.D. Thesis (University of Florida, Gainesville 2003)

52.17 A. Hyvärinen, J. Karhunen, E. Oja: *Independent Component Analysis* (Wiley, New York 2001)

52.18 A. Cichocki, S. Amari: *Adaptive Blind Signal and Image Processing* (Wiley, New York 2002)

52.19 S.C. Douglas: Blind separation of acoustic signals. In: *Microphone Arrays*, ed. by M.S. Brandstein, D.B. Ward (Springer, Berlin, Heidelberg 2001) pp. 355–380, Chap. 16

52.20 S.C. Douglas: Blind signal separation and blind deconvolution. In: *The Neural Networks for Signal Processing Handbook*, Ser. Electrical Engineering and Applied Signal Processing, ed. by Y.H. Hu, J.-N. Hwang (CRC Press, Boca Raton 2002), Chap. 7

52.21 P.D. O'Grady, B.A. Pearlmutter, S.T. Rickard: Survey of sparse and non-sparse methods in source separation, IJIST **15**, 18–33 (2005)

52.22 S. Makino, H. Sawada, R. Mukai, S. Araki: Blind source separation of convolutive mixtures of speech in frequency domain, IEICE Trans. Fundamentals **E88-A**(7), 1640–1655 (2005)

52.23 R. Lambert: Multichannel Blind Deconvolution: FIR Matrix Algebra and Separation of Multipath Mixtures. Ph.D. Thesis (University of Southern California Department of Electrical Engineering, Los Angeles 1996)

52.24 S. Roberts, R. Everson: *Independent Components Analysis: Principles and Practice* (Cambridge University Press, Cambridge 2001)

52.25 A. Gorokhov, P. Loubaton: Subspace based techniques for second order blind separation of convolutive mixtures with temporally correlated sources, IEEE Trans. Circ. Syst. **44**(9), 813–820 (1997)

52.26 H.-L. Nguyen Thi, C. Jutten: Blind source separation for convolutive mixtures, Signal Process. **45**(2), 209–229 (1995)

52.27 S. Choi, A. Cichocki: Adaptive blind separation of speech signals: Cocktail party problem, ICSP'97 (1997) pp. 617–622

52.28 S. Choi, A. Cichocki: A hybrid learning approach to blind deconvolution of linear MIMO systems, Electron. Lett. **35**(17), 1429–1430 (1999)

52.29 E. Weinstein, M. Feder, A. Oppenheim: Multichannel signal separation by decorrelation, IEEE Trans. Speech Audio Process. **1**(4), 405–413 (1993)

52.30 S.T. Neely, J.B. Allen: Invertibility of a room impulse response, J. Acoust. Soc. Am. **66**(1), 165–169 (1979)

52.31 Y.A. Huang, J. Benesty, J. Chen: Blind channel identification-based two-stage approach to separation and dereverberation of speech signals in a reverberant environment, IEEE Trans. Speech Audio Process. **13**(5), 882–895 (2005)

52.32 Y. Huang, J. Benesty, J. Chen: Identification of acoustic MIMO systems: Challenges and opportunities, Signal Process. **86**, 1278–1295 (2006)

52.33 Z. Ye, C. Chang, C. Wang, J. Zhao, F.H.Y. Chan: Blind separation of convolutive mixtures based on second order and third order statistics, Proc. ICASSP'03, Vol. 5 (2003) pp. V–305, –308

52.34 K. Rahbar, J.P. Reilly, J.H. Manton: Blind identification of MIMO FIR systems driven by quasistationary sources using second-order statistics: A frequency domain approach, IEEE Trans. Signal Process. **52**(2), 406–417 (2004)

52.35 D. Yellin, E. Weinstein: Multichannel signal separation: Methods and analysis, IEEE Trans. Signal Process. **44**(1), 106–118 (1996)

52.36 B. Chen, A.P. Petropulu: Frequency domain blind MIMO system identification based on second- and higher order statistics, IEEE Trans. Signal Process. **49**(8), 1677–1688 (2001)

52.37 B. Chen, A.P. Petropulu, L.D. Lathauwer: Blind identification of complex convolutive MIMO systems with 3 sources and 2 sensors, Proc. ICASSP'02, Vol. II (2002) pp. 1669–1672

52.38 A. Mansour, C. Jutten, P. Loubaton: Subspace method for blind separation of sources and for a convolutive mixture model, EUSIPCO 96, Signal Processing VIII, Theories and Applications (Elsevier, Amsterdam 1996) pp. 2081–2084

52.39 W. Hachem, F. Desbouvries, P. Loubaton: On the identification of certain noisy FIR convolutive mixtures, ICA'99 (1999)

52.40 A. Mansour, C. Jutten, P. Loubaton: Adaptive subspace algorithm for blind separation of independent sources in convolutive mixture, IEEE Trans. Signal Process. **48**(2), 583–586 (2000)

52.41 N. Delfosse, P. Loubaton: Adaptive blind separation of convolutive mixtures, Proc. ICASSP'96 (1996) pp. 2940–2943

52.42 N. Delfosse, P. Loubaton: Adaptive blind separation of independent sources: A second-order stable algorithm for the general case, IEEE Trans. Circ. Syst.–I: Fundamental Theory Appl. **47**(7), 1056–1071 (2000)

52.43 L.K. Hansen, M. Dyrholm: A prediction matrix approach to convolutive ICA, NNSP'03 (2003) pp. 249–258

52.44 Y. Hua, J.K. Tugnait: Blind identifiability of FIR-MIMO systems with colored input using second order statistics, IEEE Signal Process. Lett. **7**(12), 348–350 (2000)

52.45 O. Yilmaz, S. Rickard: Blind separation of speech mixtures via time-frequency masking, IEEE Trans. Signal Process. **52**(7), 1830–1847 (2004)

52.46 N. Roman: Auditory-based algorithms for sound segregatoion in multisource and reverberant environments. Ph.D. Thesis (The Ohio State University, Columbus 2005)

52.47 A. Blin, S. Araki, S. Makino: Underdetermined blind separation og convolutive mixtures of speech using time-frequency mask and mixing matrix estimation, IEICE Trans. Fundamentals **E88-A**(7), 1693–1700 (2005)

52.48 K.I. Diamantaras, A.P. Petropulu, B. Chen: Blind Two-Input-TwoOutput FIR Channel Identification based on frequency domain second- order statistics, IEEE Trans. Signal Process. **48**(2), 534–542 (2000)

52.49 E. Moulines, J.-F. Cardoso, E. Cassiat: Maximum likelihood for blind separation and deconvolution of noisy signals using mixture models, Proc. ICASSP'97, Vol. 5 (1997) pp. 3617–3620

52.50 U.A. Lindgren, H. Broman: Source separation using a criterion based on second-order statistics, IEEE Trans. Signal Process. **46**(7), 1837–1850 (1998)

52.51 H. Broman, U. Lindgren, H. Sahlin, P. Stoica: Source separation: A TITO system identification approach, Signal Process. **73**(1), 169–183 (1999)

52.52 H. Sahlin, H. Broman: MIMO signal separation for FIR channels: A criterion and performance analysis, IEEE Trans. Signal Process. **48**(3), 642–649 (2000)

52.53 C. Jutten, L. Nguyen Thi, E. Dijkstra, E. Vittoz, J. Caelen: Blind separation of sources: An algorithm for separation of convolutive mixtures. In: *Higher Order Statistics. Proceedings of the International Signal Processing Workshop*, ed. by J. Lacoume (Elsevier, Amsterdam 1992) pp. 275–278

52.54 C. Servière: Blind source separation of convolutive mixtures, SSAP'96 (1996) pp. 316–319

52.55 A. Ypma, A. Leshem, R.P. Duina: Blind separation of rotating machine sources: Bilinear forms and convolutive mixtures, Neurocomputing **49**(1–4), 349–368 (2002)

52.56 N. Charkani, Y. Deville, J. Herault: Stability analysis and optimization of time-domain convolutive source separation algorithms, SPAWC'97 (1997) pp. 73–76

52.57 N. Charkani, Y. Deville: A convolutive source separation method with self-optimizing nonlinearities, Proc. ICASSP'99, Vol. 5 (1999) pp. 2909–2912

52.58 N. Charkani, Y. Deville: Self-adaptive separation of convolutively mixed signals with a recursive structure. Part I: Stability analysis and optimization of asymptotic behaviour, Signal Process. **73**(3), 225–254 (1999)

52.59 K. Torkkola: Blind separation of convolved sources based on information maximization, NNSP'96 (1996) pp. 423–432

52.60 P. Comon: Independent component analysis, a new concept?, Signal Process. **36**(3), 287–314 (1994), special issue on Higher-Order Statistics

52.61 P. Comon, L. Rota: Blind separation of independent sources from convolutive mixtures, IEICE Trans. Fundamentals **E86-A**(3), 542–549 (2003)

52.62 V. Capdevielle, C. Servière, J.L. Lacoume: Blind separation of wide-band sources in the frequency domain, ICASSP'95, Vol. III (Detroit 1995) pp. 2080–2083

52.63 S. Icart, R. Gautier: Blind separation of convolutive mixtures using second and fourth order moments, Proc. ICASSP'96, Vol. 5 (1996) pp. 3018–3021

52.64 M. Girolami, C. Fyfe: A temporal model of linear anti-hebbian learning, Neural Process. Lett. **4**(3), 139–148 (1996)

52.65 J.K. Tugnait: On blind separation of convolutive mixtures of independent linear signals in unknown additive noise, IEEE Trans. Signal Process. **46**(11), 3117–3123 (1998)

52.66 C. Simon, P. Loubaton, C. Vignat, C. Jutten, G. d'Urso: Separation of a class of convolutive mixtures: A contrast function approach, Proc. ICASSP'99 (1999) pp. 1429–1432

52.67 Y. Su, L. He, R. Yang: An improved cumulant-based blind speech separation method, Proc. ICASSP'00 (2000) pp. 1867–1870

52.68 P. Baxter, J. McWhirter: Blind signal separation of convolutive mixtures, AsilomarSSC, Vol. 1 (2003) pp. 124–128

52.69 S. Hornillo-Mellado, C.G. Puntonet, R. Martin-Clemente, M. Rodríguez-Álvarez: Characterization of the sources in convolutive mixtures: A cumulant-based approach, ICA'04 (2004) pp. 586–593

52.70 Y. Deville, M. Benali, F. Abrard: Differential source separation for underdetermined instantaneous or convolutive mixtures: Concept and algorithms, Signal Process. **84**(10), 1759–1776 (2004)

52.71 M. Ito, M. Kawamoto, N. Ohnishi, Y. Inouye: Eigenvector algorithms with reference signals for frequency domain BSS, ICA'06 (2006) pp. 123–131

52.72 W. Baumann, D. Kolossa, R. Orglmeister: Beamforming-based convolutive source separation, Proc. ICASSP'03, Vol. V (2003) pp. 357–360

52.73 W. Baumann, D. Kolossa, R. Orglmeister: Maximum likelihood permutation correction for convolutive source separation, ICA'03 (2003) pp. 373–378

52.74 M.S. Pedersen, C.M. Nielsen: Gradient flow convolutive blind source separation, MLSP'04 (2004) pp. 335–344

52.75 J.-F. Cardoso, A. Souloumiac: Blind beamforming for non Gaussian signals, IEEE Proc. **140**(6), 362–370 (1993)

52.76 D. Yellin, E. Weinstein: Criteria for multichannel signal separation, IEEE Trans. Signal Process. **42**(8), 2158–2168 (1994)

52.77 D. Kolossa, R. Orglmeister: Nonlinear postprocessing for blind speech separation, ICA'04 (2004) pp. 832–839

52.78 P. Comon, E. Moreau, L. Rota: Blind separation of convolutive mixtures: A contrast-based joint diagonalization approach, ICA'01 (2001) pp. 686–691

52.79 E. Moreau, J. Pesquet: Generalized contrasts for multichannel blind deconvolution of linear systems, IEEE Signal Process. Lett. **4**(6), 182–183 (1997)

52.80 Y. Li, J. Wang, A. Cichocki: Blind source extraction from convolutive mixtures in ill-conditioned multi-input multi-output channels, IEEE Trans. Circ. Syst. I: Regular Papers **51**(9), 1814–1822 (2004)

52.81 R.K. Prasad, H. Saruwatari, K. Shikano: Problems in blind separation of convolutive speech mixtures by negentropy maximization, IWAENC'03 (2003) pp. 287–290

52.82 X. Sun, S. Douglas: Adaptive paraunitary filter banks for contrast-based multichannel blind deconvolution, Proc. ICASSP'01, Vol. 5 (2001) pp. 2753–2756

52.83 J. Thomas, Y. Deville, S. Hosseini: Time-domain fast fixed-point algorithms for convolutive ICA, IEEE Signal Process. Lett. **13**(4), 228–231 (2006)

52.84 C. Jutten, J. Herault: Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture, Signal Process. **24**(1), 1–10 (1991)

52.85 A.D. Back, A.C. Tsoi: Blind deconvolution of signals using a complex recurrent network, NNSP'94 (1994) pp. 565–574

52.86 I. Kopriva, Ž. Devčić, H. Szu: An adaptive short-time frequency domain algorithm for blind separation of nonstationary convolved mixtures, IJCNN'01 (2001) pp. 424–429

52.87 S. Cruces, L. Castedo: A Gauss-Newton method for blind source separation of convolutive mixtures, Proc. ICASSP'98, Vol. IV (1998) pp. 2093–2096

52.88 S.V. Gerven, D.V. Compernolle: Signal separation by symmetric adaptive decorrelation: Stability, convergence, and uniqueness, IEEE Trans. Signal Process. **43**(7), 1602–1612 (1995)

52.89 S. Li, T.J. Sejnowski: Adaptive separation of mixed broadband sound sources with delays by a beamforming Herault-Jutten network, IEEE J. Ocean. Eng. **20**(1), 73–79 (1995)

52.90 N. Charkani, Y. Deville: Optimization of the asymptpotic preformance of time-domain convolutive source separation algorithms, ESANN'97 (1997) pp. 273–278

52.91 N. Charkani, Y. Deville: Self-adaptive separation of convolutively mixed signals with a recursive structure. Part II: Theoretical extensions and application to synthetic and real signals, Signal Process. **75**(2), 117–140 (1999)

52.92 S. Choi, H. Hong, H. Glotin, F. Berthommier: Multichannel signal separation for cocktail party speech recognition: A dynamic recurrent network, Neurocomputing **49**(1–4), 299–314 (2002)

52.93 F. Berthommier, S. Choi: Several improvements of the Hérault-Jutten model for speech segregation, ICA'03 (2003) pp. 1089–1094

52.94 M. Cohen, G. Cauwenbergh: Blind separation of linear convolutive mixtures through parallel stochastic optimization, ISCAS'98, Vol. 3 (1998) pp. 17–20

52.95 M. Stanacevic, M. Cohen, G. Cauwenberghs: Blind separation of linear convolutive mixtures using orthogonal filter banks, ICA'01 (2001) pp. 260–265

52.96 G. Deco, D. Obradovic: *An Information-Theoretic Approach to Neural Computing* (Springer, New York 1996)

52.97 A.J. Bell, T.J. Sejnowski: An information-maximization approach to blind separation and blind deconvolution, Neural Comput. **7**(6), 1129–1159 (1995)

52.98 S. Amari, S. Douglas, A. Cichocki, H.H. Yang: Multichannel blind deconvolution and equalization using the natural gradient, IEEE International Workshop on Wireless Communication (1997) pp. 101–104

52.99 B.A. Pearlmutter, L.C. Parra: Maximum likelihood blind source separation: A context-sensitive generalization of ICA, NIPS'97 (1997) pp. 613–619

52.100 J.-F. Cardoso: Blind signal separation: Statistical principles, Proc. IEEE **9**(10), 2009–2025 (1998)
52.101 K. Kokkanikis, A.K. Nandi: Optimal blind separation of convolutive audio mixtures without temporal constraints, Proc. ICASSP'04, Vol. I (2004) pp. 217–220
52.102 K. Kokkinakis, A.K. Nandi: Multichannel speech separation using adaptive parameterization of source pdfs, ICA'04 (2004) pp. 486–493
52.103 H.-C. Wu, J.C. Principe: Generalized anti-hebbian learning for source separation, Proc. ICASSP'99, Vol. 2 (1999) pp. 1073–1076
52.104 J. Xi, J.P. Reilly: Blind separation and restoration of signals mixed in convolutive environment, Proc. ICASSP'97 (1997) pp. 1327–1330
52.105 J.P. Reilly, L.C. Mendoza: Blind signal separation for convolutive mixing environments using spatial-temporal processing, Proc. ICASSP'99 (1999) pp. 1437–1440
52.106 H. Sawada, R. Mukai, S. Araki, S. Makino: A robust and precise method for solving the permutation problem of frequency-domain blind source separation, IEEE Trans. Speech Audio Process. **12**(5), 530–538 (2004)
52.107 H. Sawada, R. Mukai, S. Araki, S. Makino: Polar coordinate based nonlinear function for frequency domain blind source separation, Proc. ICASSP'02 (2002) pp. 1001–1004
52.108 J.-F. Cardoso, T. Adali: The maximum likelihood approach to complex ICA, Proc. ICASSP, Vol. V (2006) pp. 673–676
52.109 M. Novey, T. Adali: Adaptable nonlinearity for complex maximization of nongaussianity and a fixed-point algorithm, MLSP (2006)
52.110 M. Joho, H. Mathis, G.S. Moschytz: An FFT-based algorithm for multichannel blind deconvolution, ISCAS'99, Vol. 3 (1999) pp. 203–206
52.111 R.H. Lambert, A.J. Bell: Blind separation of multiple speakers in a multipath environment, Proc. ICASSP'97, Vol. 1 (1997) pp. 423–426
52.112 R.H. Lambert: A new method for source separation, Proc. ICASSP'95, Vol. 3 (1995) pp. 2116–2119
52.113 T.-W. Lee, A.J. Bell, R. Orglmeister: Blind source separation of real world signals, ICNN'97 (1997) pp. 2129–2135
52.114 S. Choi, S. Amari, A. Cichocki, R. Liu: Natural gradient learning with a nonholonomic constraint for blind deconvolution of multiple channels, ICA'99 (1999) pp. 371–376
52.115 S.C. Douglas, X. Sun: Convolutive blind separation of speech mixtures using the natural gradient, Speech Commun. **39**, 65–78 (2003)
52.116 K. Matsuoka, Y. Ohba, Y. Toyota, S. Nakashima: Blind separation for convolutive mixture of many voices, IWAENC'03 (2003) pp. 279–282
52.117 K. Matsuoka, S. Nakashima: Minimal distortion principle for blind source separation, ICA'01 (2001) pp. 722–727
52.118 W. Wang, M.G. Jafari, S. Sanei, J.A. Chambers: Blind separartion of convolutive mixtures of cyclostationary signals, Int. J. Adapt. Contr. Signal Process. **18**, 279–298 (2004)
52.119 G.-J. Jang, C. Choi, Y. Lee, Y.-H. Oh: Adaptive cross-channel interference cancellation on blind signal separation outputs using source absence/presence detection and spectral subtraction, ICLSP'04, Vol. IV (2004) pp. 2865–2868
52.120 S.H. Nam, S. Beack: A frequency-domain normalized multichannel blind deconvolution algorithm for acoustical signals, ICA'04 (2004) pp. 524–531
52.121 M. Joho, P. Schniter: Frequency domain realization of a multichannel blind deconvolution algorithm based on the natural gradient, ICA'03 (2003) pp. 543–548
52.122 C. Choi, G. Jang, Y. Lee, S.R. Kim: Adaptive cross-channel interference cancellation on blind source separation outputs, ICA'04 (2004) pp. 857–864
52.123 L. Parra, C. Spence, B. de Vries: Convolutive source separation and signal modeling with ML, ISIS'97 (1997)
52.124 L.C. Parra: Temporal models in blind source separation, Lect. Notes Comput. Sci. **1387**, 229–247 (1998)
52.125 K. Yamamoto, F. Asano, W. van Rooijen, E. Ling, T. Yamada, N. Kitawaki: Estimation of the number of sound sources using support vector machines and its application to sound source separation, Proc. ICASSP'03, Vol. 5 (2003) pp. 485–488
52.126 J. Anemüller: Across-frequency processing in convolutive blind source separation. Ph.D. Thesis (Univ. Oldenburg, Oldenburg 2001)
52.127 J. Anemüller, T. Gramms: On-line blind separation of moving sound sources, ICA'99 (1999) pp. 331–334
52.128 S. Deligne, R. Gopinath: An EM algorithm for convolutive independent component analysis, Neurocomputing **49**, 187–211 (2002)
52.129 J. Rosca, C. Borss, R. Balan: Generalized sparse signal mixing model and application to noisy blind source separation, Proc. ICASSP'04, Vol. III (2004) pp. 877–880
52.130 S.C. Douglas, H. Sawada, S. Makino: Natural gradient multichannel blind deconvolution and speech separation using causal FIR filters, IEEE Trans. Speech Audio Process. **13**(1), 92–104 (2005)
52.131 S. Araki, R. Mukai, S. Makino, T. Nishikawa, H. Saruwatari: The fundamental limitation of frequency domain blind source separation for convolutive mixtures of speech, IEEE Trans. Speech Audio Process. **11**(2), 109–116 (2003)
52.132 S. Ukai, T. Takatani, T. Nishikawa, H. Saruwatari: Blind source separation combining SIMO-model-based ICA and adaptive beamforming, Proc. ICASSP'05, Vol. III (2005) pp. 85–88

52.133 R.H. Lambert, C.L. Nikias: Polynomial matrix whitening and application to the multichannel blind deconvolution problem, MILCOM'95, Vol. 3 (1995) pp. 988–992

52.134 N. Mitianoudis, M. Davies: Audio source separation of convolutive mixtures, IEEE Trans. Speech Audio Process. **11**(5), 489–497 (2002)

52.135 F. Tordini, F. Piazza: A semi-blind approach to the separation of real world speech mixtures, IJCNN'02, Vol. 2 (2002) pp. 1293–1298

52.136 H. Attias: Source separation with a sensor array using graphical models and subband filtering, NIPS'02, Vol. 15 (2002) pp. 1229–1236

52.137 M.A. Gandhi, M.A. Hasegawa-Johnson: Source separation using particle filters, ICLSP'04, Vol. III (2004) pp. 2449–2452

52.138 R.K. Olsson, L.K. Hansen: A harmonic excitation state-space approach to blind separation of speech, NIPS (2004)

52.139 R.K. Olsson, L.K. Hansen: Probabilistic blind deconvolution of non-stationary sources, EUSIPCO'04 (2004) pp. 1697–1700

52.140 S. Winter, H. Sawada, S. Araki, S. Makino: Hierarchical clustering applied to overcomplete BSS for convolutive mixtures, SAPA'04 (2004)

52.141 S. Winter, H. Sawada, S. Araki, S. Makino: Overcomplete BSS for convolutive mixtures based on hierarchical clustering, ICA'04 (2004) pp. 652–660

52.142 H. Attias: New EM algorithms for source separation and deconvolution, Proc. ICASSP'03, Vol. V (2003) pp. 297–300

52.143 C. Andrieu, S. Godsill: A particle filter for model based audio source separation, ICA'00 (2000) pp. 381–386

52.144 J.R. Hopgood: Bayesian blind MIMO deconvolution of nonstationary subband autoregressive sources mixed through subband all-pole channels, SSP'03 (2003) pp. 422–425

52.145 S.J. Godsill, C. Andrieu: Bayesian separation and recovery of convolutively mixed autoregressive sources, Proc. ICASSP'99, Vol. III (1999) pp. 1733–1736

52.146 K. Abed-Meraim, W. Qiu, Y. Hua: Blind system identification, Proc. IEEE **85**(8), 1310–1322 (1997)

52.147 S. Sanei, W. Wang, J.A. Chambers: A coupled HMM for solving the permutation problem in frequency domain BSS, Proc. ICASSP'04, Vol. V (2004) pp. 565–568

52.148 W. Wang, D. Cosker, Y. Hicks, S. Sanei, J. Chambers: Video assisted speech source seperation, Proc. ICASSP'05, Vol. V (2005) pp. 425–428

52.149 L. Parra, C. Spence: Convolutive blind separation of non-stationary sources, IEEE Trans. Speech Audio Process. **8**(3), 320–327 (2000)

52.150 D.W.E. Schobben, P.C.W. Sommen: On the indeterminacies of convolutive blind signal separation based on second-order statistics, ISSPA'99 (1999) pp. 215–218

52.151 J. Liang, Z. Ding: Blind MIMO system identification based on cumulant subspace decomposition, IEEE Trans. Signal Process. **51**(6), 1457–1468 (2003)

52.152 K. Rahbar, J.P. Reilly: A frequency domain method for blind source separation of convolutive audio mixtures, IEEE Trans. Speech Audio Process. **13**(5), 832–844 (2005)

52.153 M. Kawamoto, K. Matsuoka, N. Ohnishi: A method of blind separation for convolved non-stationary signals, Neurocomputing **22**(1–3), 157–171 (1998)

52.154 A. Belouchrani, K. Abed-Meraim, J.-F. Cardoso, E. Moulines: A blind source separation technique using second-order statistics, IEEE Trans. Signal Process. **45**(2), 434–444 (1997)

52.155 A.G. Lindgren, T.P. von Hoff, A.N. Kaelin: Stability and performance of adaptive algorithms for multichannel blind source separation and deconvolution, EUSIPCO'00, Vol. 2 (2000) pp. 861–864

52.156 H. Sahlin, H. Broman: Separation of real-world signals, Signal Process. **64**, 103–113 (1998)

52.157 D.C.B. Chan, P.J.W. Rayner, S.J. Godsill: Multichannel blind signal separation, Proc. ICASSP'96 (1996) pp. 649–652

52.158 C. Simon, C. Vignat, P. Loubaton, C. Jutten, G. d'Urso: On the convolutive mixture – source separation by the decorrelation approach, Proc. ICASSP'98, Vol. 4 (1998) pp. 2109–2112

52.159 N. Murata, S. Ikeda, A. Ziehe: *An Approach to Blind Source Separation Based on Temporal Structure of Speech Signals* (RIKEN Brain Science Institute, Japan 1998), BSIS Technical Reports 98-2

52.160 B. Yin, P. Sommen: Adaptive blind signal separation using a new simplified mixing model, ProRISC'99, ed. by J. Veen (1999) pp. 601–606

52.161 H.-C. Wu, J.C. Principe: Simultaneous diagonalization in the frequency domain (SDIF) for source separation, ICA'99 (1999) pp. 245–250

52.162 A. Souloumiac: Blind source detection and separation using second order non-stationarity, Proc. ICASSP'95, Vol. III (1995) pp. 1912–1915

52.163 K. Matsuoka, M. Ohya, M. Kawamoto: A neural net for blind separation of nonstationary signals, Neural Netw. **8**(3), 411–419 (1995)

52.164 M. Kawamoto, A.K. Barros, A. Mansour, K. Matsuoka, N. Ohnishi: Blind separation for convolutive mixtures of non-stationary signals, Int. Conf. Neural Inf. Process. (1998) pp. 743–746

52.165 M. Kawamoto, A.K. Barros, K. Matsuoka, N. Ohnishi: A method of real-world blind separation implemented in frequency domain, ICA'00 (2000) pp. 267–272

52.166 M. Ito, M. Maruyoshi, M. Kawamoto, T. Mukai, N. Ohneshi: Effectiveness of directional microphones and utilization of source arriving directions in source separation, ICONIP'02 (2002) pp. 523–526

52.167 M. Ito, Y. Takeuchi, T. Matsumoto, H. Kudo, M. Kawamoto, T. Mukai, N. Ohnishi: Moving-source separation using directional microphones, ISSPIT'02 (2002) pp. 523–526
52.168 Y. Katayama, M. Ito, Y. Takeuchi, T. Matsumoto, H. Kudo, N. Ohnishi, T. Mukai: Reduction of source separation time by placing microphones close together, ISSPIT'02 (2002) pp. 540–544
52.169 R. Aichner, S. Araki, S. Makino, T. Nishikawa, H. Saruwatari: Time domain blind source separation of non-stationary convolved signals by utilizing geometric beamforming, NNSP'02 (2002) pp. 445–454
52.170 R. Aichner, H. Buchner, S. Araki, S. Makino: On-line time-domain blind source separation of nonstationary convolved signals, ICA'03 (2003) pp. 987–992
52.171 H. Buchner, R. Aichner, W. Kellermann: A generalization of blind source separation algorithms for convolutive mixtures based on second-order statistics, IEEE Trans. Speech Audio Process. **13**(1), 120–134 (2005)
52.172 E. Visser, T.-W. Lee: Speech enhancement using blind source separation and two-channel energy based speaker detection, Proc. ICASSP'03, Vol. 1 (2003) pp. 884–887
52.173 E. Visser, K. Chan, S. Kim, T.-W. Lee: A comparison of simultaneous 3-channel blind source separation to selective separation on channel pairs using 2-channel BSS, ICLSP'04, Vol. IV (2004) pp. 2869–2872
52.174 M.Z. Ikram, D.R. Morgan: A multiresolution approach to blind separation of speech signals in a reverberant environment, Proc. ICASSP'01, Vol. V (2001)
52.175 M. Ikram, D. Morgan: A beamforming approach to permutation alignment for multichannel frequency-domain blind speech separation, Proc. ICASSP'02 (2002) pp. 881–884
52.176 M.Z. Ikram, D.R. Morgan: Permutation inconsistency in blind speech separation: Investigation and solutions, IEEE Trans. Speech Audio Process. **13**(1), 1–13 (2005)
52.177 L. Parra, C. Spence: On-line convolutive source separation of non-stationary signals, IEEE J. VLSI Signal Process. **26**(1,2), 39–46 (2000)
52.178 M. Joho: Blind signal separation of convolutive mixtures: A time-domain joint-diagonalization approach, ICA'04 (2004) pp. 578–585
52.179 M. Joho: Convolutive blind signal separation in acoustics by joint approximate diagonalization of spatiotemporal correlation matrices, Asilomar SSC (2004)
52.180 S. Araki, S. Makino, R. Mukai, H. Saruwatari: Equivalence between frequency domain blind source separation and frequency domain adaptive beamformers, CRAC'01 (2001)
52.181 C.L. Fancourt, L. Parra: The coherence function in blind source separation of convolutive mixtures of non-stationary signals, NNSP (2001) pp. 303–312
52.182 C.L. Fancourt, L. Parra: The generalized sidelobe decorrelator, WASPAA'01 (2001)
52.183 C. Fancourt, L. Parra: A comparison of decorrelation criteria for the blind source separation of non-stationary signals, SAM'02 (2002)
52.184 L. Parra, C. Alvino: Geometric source separation: Merging convolutive source separation with geometric beamforming, IEEE Trans. Speech Audio Process. **10**(6), 352–362 (2002)
52.185 L. Parra, C. Fancourt: An Adaptive Beamforming Perspective on Convolutive Blind Source Separation. In: *Noise Reduction in Speech Applications*, ed. by G. Davis (CRC, Boca Raton 2002)
52.186 E. Visser, T.-W. Lee: Blind source sepoaration in mobile environments usinbg a priori knowledge, Proc. ICASSP'04, Vol. III (2004) pp. 893–896
52.187 M.S. Pedersen, L.K. Hansen, U. Kjems, K.B. Rasmussen: Semi-blind source separation using head-related transfere functions, Proc. ICASSP'04, Vol. V (2004) pp. 713–716
52.188 S. Ding, T. Hikichi, T. Niitsuma, M. Hamatsu, K. Sugai: Recursive method for blind source separation and its applications to real-time separations of acoustic signals, ICA'03 (2003) pp. 517–522
52.189 E. Robledo-Arnuncio, B.F. Juang: Issues in frequency domain blind source separation – a critical revisit, Proc. ICASSP'05, Vol. V (2005) pp. 281–284
52.190 W. Wang, J.A. Chambers, S. Sanei: A joint diagonalization method for convolutive blind separation of nonstationary sources in the frequency domain, ICA'03 (2003) pp. 939–944
52.191 W. Wang, J.A. Chambers, S. Sanei: A novel hybrid approach to the permutation problem of frequency domain blind source separation, ICA'04 (2004) pp. 532–539
52.192 W. Wang, S. Sanei, J.A. Chambers: Penalty function-based joint diagonalization approach for convolutive blind separation of nonstationary sources, IEEE Trans. Signal Process. **53**(5), 1654–1669 (2005)
52.193 L. Molgedey, H. Schuster: Separation of independent signals using time-delayed correlations, Phys. Rev. Lett. **72**(23), 3634–3637 (1994)
52.194 T.J. Ngo, N. Bhadkamkar: Adaptive blind separation of audio sources by a physically compact device using second order statistics, ICA'99 (1999) pp. 257–260
52.195 D.W. Schobben, P.C.W. Sommen: A frequency domain blind signal separation method based on decorrelation, IEEE Trans. Signal Process. **8**(50), 1855–1865 (2002)
52.196 S. Ikeda, N. Murata: An approach to blind source separation of speech signals, ICANN'98, Vol. 2 (1998) pp. 761–766

52.197 S. Ikeda, N. Murata: A method of blind separation on temporal structre of signals, ICONIP'98, Vol. 2 (1998) pp. 737–742

52.198 S. Ikeda, N. Murata: A method of ICA in time-frequency domain, ICA'99 (1999) pp. 365–371

52.199 N. Murata, S. Ikeda, A. Ziehe: An approach to blind source separation based on temporal structure of speech signals, Neurocomputing **41**(1–4), 1–24 (2001)

52.200 A. Ahmed, P.J.W. Rayner, S.J. Godsill: Considering non-stationarity for blind signal separation, WASPAA'99 (1999) pp. 111–114

52.201 B. Yin, P. Sommen: A new convolutive blind signal separation algorithm based on second order statistics using a simplified mixing model, EUSIPCO'00, Vol. 4 (2000) pp. 2049–2053

52.202 D.-T. Pham, C. Servière, H. Boumaraf: Blind separation of convolutive audio mixtures using nonstationarity, ICA'03 (2003) pp. 981–986

52.203 C. Servière, D. Pham: A novel method for permutation correction in frequency-domain in blind separation of speech mixtures, ICA'04 (2004) pp. 807–815

52.204 H. Buchner, R. Aichner, W. Kellermann: Blind source separation for convolutive mixtures exploiting nongaussianity, nonwhiteness, and nonstationarity, IWAENC'03 (2003) pp. 275–278

52.205 R. Aichner, H. Buchner, F. Yan, W. Kellermann: Real-time convolutive blind source separation based on a broadband approach, ICA'04 (2004) pp. 840–848

52.206 H. Buchner, R. Aichner, W. Kellermann: Trinicon: A versatile framework for multichannel blind signal processing, Proc. ICASSP'04, Vol. III (2004) pp. 889–892

52.207 R. Aichner, H. Buchner, W. Kellermann: On the causality problem in time-domain blind source separation and deconvolution algorithms, Proc. ICASSP'05, Vol. V (2005) pp. 181–184

52.208 B.S. Krongold, D.L. Jones: Blind source separation of nonstationary convolutively mixed signals, SSAP'00 (2000) pp. 53–57

52.209 K. Rahbar, J.P. Reilly: Blind source separation algorithm for MIMO convolutive mixtures, ICA'01 (2001)

52.210 A. Holobar, D. Zazula: A novel approach to convolutive blind separation of close-to-orthogonal pulse sources using second-order statistics, EUSIPCO'04 (2004) pp. 381–384

52.211 K.-C. Yen, Y. Zhao: Adaptive co-channel speech separation and recognition, IEEE Trans Speech Audio Process. **7**(2), 138–151 (1999)

52.212 A. Mertins, I. Russel: An extended ACDC algorithm for the blind estimation of convolutive mixing systems, ISSPA'03, Vol. 2 (2003) pp. 527–530

52.213 Y. Zhao, R. Hu: Fast convergence speech source separation in reverberant acoustic environment, Proc. ICASSP'04, Vol. III (2004) pp. 897–900

52.214 I. Russell, J. Xi, A. Mertins, J. Chicharo: Blind source separation of nonstationary convolutively mixed signals in the subband domain, Proc. ICASSP'04, Vol. V (2004) pp. 484–484

52.215 A. Leon-Garcia: *Probability, Random Processes for Electrical Engineering*, 2nd edn. (Addison Wesley, Reading 1994)

52.216 S. Shamsunder, G.B. Giannakis: Multichannel blind signal separation and reconstruction, IEEE Trans. Speech, Audio Process. **5**(6), 515–528 (1997)

52.217 L. Deneire, D.T. Slock: A Schur method for multiuser multichannel blind identification, Proc. ICASSP'99 (1999) pp. 2905–2908

52.218 C.T. Ma, Z. Ding, S.F. Yau: A two-stage algorithm for MIMO blind deconvolution of nonstationary colored signals, IEEE Trans. Signal Process. **48**(4), 1187–1192 (2000)

52.219 I. Bradaric, A.P. Petropulu, K.I. Diamantaras: On blind identifiability of FIR-MIMO systems with cyclostationary inputs using second order statistics, Proc. ICASSP'02, Vol. II (2002) pp. 1745–1748

52.220 I. Bradaric, A.P. Petropulu, K.I. Diamantaras: On blind identifiability of FIR-MIMO systems with cyclostationary inputs using second order statistics, IEEE Trans. Signal Process. **51**(2), 434–441 (2003)

52.221 M. Knaak, M. Kunter, D. Filbert: Blind source separation for acoustical machine diagnosis, DSP'02 (2002)

52.222 M. Knaak, S. Araki, S. Makino: Geometrically constrained ICA for robust separation of sound mixtures, ICA'03 (2003) pp. 951–956

52.223 T. Mei, F. Yin: Blind separation of convolutive mixtures by decorrelation, Signal Process. **84**(12), 2297 (2004)

52.224 S. Rickard, O. Yilmaz: On the approximate W-disjoint orthogonality of speech, Proc. ICASSP'02, Vol. I (2002) pp. 529–532

52.225 S. Rickard, T. Melia, C. Fearon: DESPRIT – histogram based blind source separation of more sources than sensors using subspace methods, WASPAA'05 (2005) pp. 5–8

52.226 A. Jourjine, S. Rickard, O. Yilmaz: Blind separation of disjoint orthogonal signals: Demixing N sources from 2 mixtures, Proc. ICASSP'00, Vol. V (2000) pp. 2985–2988

52.227 S. Araki, S. Makino, H. Sawada, R. Mukai: Reducing musical noise by a fine-shift overlap-add method applied to source separation using a time-frequency mask, Proc. ICASSP'05, Vol. III (2005) pp. 81–84

52.228 M.S. Pedersen, D. Wang, J. Larsen, U. Kjems: Overcomplete blind source separation by combining ICA and binary time-frequency masking, MLSP'05 (2005)

52.229 M.S. Pedersen, D.L. Wang, J. Larsen, U. Kjems: Separating underdetermined convolutive speech mixtures, ICA'06 (2006) pp. 674–681

52.230 H. Sawada, S. Araki, R. Mukai, S. Makino: Blind extraction of a dominant source signal from mixtures of many sources, Proc. ICASSP'05, Vol. III (2005) pp. 61–64
52.231 H.-C. Wu, J.C. Principe, D. Xu: Exploring the time-frequency microstructure of speech for blind source separation, Proc. ICASSP'98, Vol. 2 (1998) pp. 1145–1148
52.232 J.M. Peterson, S. Kadambe: A probabilistic approach for blind source separation of underdetermined convolutive mixtures, Proc. ICASSP'03, Vol. 6 (2003) pp. 581–584
52.233 D. Luengo, I. Santamaria, L. Vielva, C. Pantaleon: Underdetermined blind separation of sparse sources with instantaneous and convolutive mixtures, NNSP'03 (2003) pp. 279–288
52.234 S.A. Abdallah, M.D. Plumbley: Application of geometric dependency analysis to the seperation of convolved mixtures, ICA'04 (2004) pp. 540–547
52.235 M. Babaie-Zadeh, A. Mansour, C. Jutten, F. Marvasti: A geometric approach for separating several speech signals, ICA'04 (2004) pp. 798–806
52.236 Y. Li, A. Cichocki, L. Zhang: Blind source estimation of FIR channels for binary sources: A grouping decision approach, Signal Process. **84**(12), 2245–2263 (2004)
52.237 B.A. Pearlmutter, A.M. Zador: Monaural source separation using spectral cues, ICA'04 (2004) pp. 478–485
52.238 P. Smaragdis: Non negative matrix factor deconvolution, extraction of multiple sound sources from monophonic inputs, ICA'04 (2004) pp. 494–499
52.239 T. Virtanen: Separation of sound sources by convolutive sparse coding, SAPA'04 (2004)
52.240 M.S. Pedersen, T. Lehn-Schiøler, J. Larsen: BLUES from music: BLind Underdetermined Extraction of Sources from Music, ICA'06 (2006) pp. 392–399
52.241 A.S. Bregman: *Auditory Scene Analysis*, 2nd edn. (MIT Press, Cambridge 1990)
52.242 M. Weintraub: The GRASP sound separation system, Proc. ICASSP'84 (1984) pp. 69–72
52.243 R.R. Guddeti, B. Mulgrew: Perceptually motivated blind source separation of convolutive mixtures, Proc. ICASSP'05, Vol. V (2005) pp. 273–276
52.244 A.K. Barros, T. Rutkowski, F. Itakura, N. Ohnishi: Estimation of speech embedded in a reverberant and noisy environment by independent component analysis and wavelets, IEEE Trans. Neural Netw. **13**(4), 888–893 (2002)
52.245 M. Furukawa, Y. Hioka, T. Ema, N. Hamada: Introducing new mechanism in the learning process of FDICA-based speech separation, IWAENC'03 (2003) pp. 291–294
52.246 R.D. Patterson: The sound of a sinusoid: Spectral models, J. Acoust Soc. Am. **96**, 1409–1418 (1994)
52.247 T. Rutkowski, A. Cichocki, A.K. Barros: Speech enhancement from interfering sounds using CASA techniques and blind source separation, ICA'01 (2001) pp. 728–733
52.248 N. Roman, D. Wang, G.J. Brown: Speech segregation based on sound localization, J. Acoust. Soc. Am. **114**(4), 2236–2252 (2003)
52.249 T. Nishikawa, H. Saruwatari, K. Shikano: Blind source separation of acoustic signals based on multistage ICA combining frequency-domain ICA and time-domain ICA, IEICE Trans. Fundamentals **E86-A**(4), 846–858 (2003)
52.250 P. Smaragdis: Efficient blind separation of convolved sound mixtures, WASPAA'97 (1997)
52.251 M. Davies: Audio source separation. In: *Mathematics in Signal Processing V*, ed. by J.G. McWhirter, I.K. Proudler (Oxford Univ. Press, Oxford 2001)
52.252 F. Duplessis-Beaulieu, B. Champagne: Fast convolutive blind speech separation via subband adaptation, Proc. ICASSP'03, Vol. 5 (2003) pp. 513–516
52.253 C. Servière: Separation of speech signals under reverberant conditions, EUSIPCO'04 (2004) pp. 1693–1696
52.254 T.-W. Lee, A.J. Bell, R.H. Lambert: Blind separation of delayed and convolved sources, NIPS, Vol. 9 (1997) pp. 758–764
52.255 T.-W. Lee, A. Ziehe, R. Orglmeister, T.J. Sejnowski: Combining time-delayed decorrelation and ICA: towards solving the cocktail party problem, Proc. ICASSP'98, Vol. 2 (1998) pp. 1249–1252
52.256 A. Westner, V.M. Bove Jr.: Blind separation of real world audio signals using overdetermined mixtures, ICA'99 (1999)
52.257 K. Kokkinakis, A.K. Nandi: Multichannel blind deconvolution for source separation in convolutive mixtures of speech, IEEE Trans. Audio Speech Lang. Process. **14**(1), 200–212 (2006)
52.258 H. Sawada, R. Mukai, S. de la Kethulle, S. Araki, S. Makino: Spectral smoothing for frequency-domain blind source separation, IWAENC'03 (2003) pp. 311–314
52.259 H. Sawada, R. Mukai, S. Araki, S. Makino: Convolutive blind source separation for more than two sources in the frequency domain, Proc. ICASSP'04, Vol. III (2004) pp. 885–888
52.260 D.W.E. Schobben, P.C.W. Sommen: A new blind signal separation algorithm based on second-order statistics, IASTED SIP'06 (1998) pp. 564–569
52.261 H. Attias, J.C. Platt, A. Acero, L. Deng: Speech denoising and dereverberation using probabilistic models, NIPS'01, Vol. 13 (2001)
52.262 R. Aichner, H. Buchner, W. Kellermann: A novel normalization and regularization scheme for broadband convolutive blind source separation, ICA'06 (2006) pp. 527–535
52.263 H. Sawada, S. Araki, R. Mukai, S. Makino: Blind source separation with different sensor spacing and filter length for each frequency range, NNSP'02 (2002) pp. 465–474

52.264 P. Smaragdis: Blind separation of convolved mixtures in the frequency domain, Neurocomputing **22**(1–3), 21–34 (1998)
52.265 V.C. Soon, L. Tong, Y.F. Huang, R. Liu: A wideband blind identification approach to speech acquisition using a microphone array, Proc. ICASSP'92, Vol. 1 (1992) pp. 293–296
52.266 S. Kurita, H. Saruwatari, S. Kajita, K. Takeda, F. Itakura: Evaluation of frequency-domain blind signal separation using directivity pattern under reverberant conditions, Proc. ICASSP'00 (2000) pp. 3140–3143
52.267 F. Asano, S. Ikeda, M. Ogawa, H. Asoh, N. Kitawaki: Combined approach of array processing and independent component analysis for blind separation of acoustic signals, IEEE Trans. Speech Audio Process. **11**(3), 204–215 (2003)
52.268 N. Mitianoudis, M.E. Davies: Permutation alignment for frequency domain ICA using subspace beamforming methods, ICA'04 (2004) pp. 669–676
52.269 W. Baumann, B.-U. Köhler, D. Kolossa, R. Orglmeister: Real time separation of convolutive mixtures, ICA'01 (2001) pp. 65–69
52.270 H. Gotanda, K. Nobu, T. Koya, K. Kaneda, T. Ishibashi, N. Haratani: Permutation correction and speech extraction based on split spectrum through fastICA, ICA'03 (2003) pp. 379–384
52.271 S. Araki, S. Makino, R. Aichner, T. Nishikawa, H. Saruwatari: Subband based blind source separation with appropriate processing for each frequency band, ICA'03 (2003) pp. 499–504
52.272 J. Anemüller, B. Kollmeier: Amplitude modulation decorrelation for convolutive blind sourc separation, ICA'00 (2000) pp. 215–220
52.273 J. Antoni, F. Guillet, M. El Badaoui, F. Bonnardot: Blind separation of convolved cyclostationary processes, Signal Process. **85**(1), 51–66 (2005)
52.274 A. Dapena, C. Serviere: A simplified frequency-domain approach for blind separation of convolutive mixtures, ICA'01 (2001) pp. 569–574
52.275 C. Mejuto, A. Dapena, L. Castedo: Frequency-domain infomax for blind separation of convolutive mixtures, ICA'00 (2000) pp. 315–320
52.276 N. Mitianoudis, M. Davies: New fixed-point ICA algorithms for convolved mixtures, ICA'01 (2001) pp. 633–638
52.277 I. Lee, T. Kim, T.-W. Lee: Complex fastiva: A robust maximum likelihood approach of mica for convolutive bss, ICA'06 (2006) pp. 625–632
52.278 T. Kim, H. Attias, S.-Y. Lee, T.-W. Lee: Blind source separation exploiting higher-order frequency dependencies, IEEE Trans. Audio Speech Lang. Process. **15**(1), 70–79 (2006)
52.279 S. Dubnov, J. Tabrikain, M. Arnon-Targan: A method for directionally-disjoint source separation in convolutive environment, Proc. ICASSP'04, Vol. V (2004) pp. 489–492
52.280 A. Hiroe: Solution of permutation problem in frequency domain ica, using multivariate probability density functions, ICA'06 (2006) pp. 601–608
52.281 D. Kolossa, B.-U. Köhler, M. Conrath, R. Orgelmeister: Optimal permutation correlation by multiobjective genetic algorithms, ICA'01 (2001) pp. 373–378
52.282 K. Kamata, X. Hu, H. Kobataka: A new approach to the permutation problem in frequency domain blind source separation, ICA'04 (2004) pp. 849–856
52.283 H. Attias, C.E. Schreiner: Blind source separation and deconvolution: The dynamic component analysis algorithm, Neural Comput. **11**, 803–852 (1998)
52.284 F. Asano, S. Ikeda, M. Ogawa, H. Asoh, N. Kitawaki: Blind source separation in reflective sound fields, HSC'01 (2001)
52.285 H. Sawada, S. Araki, R. Mukai, S. Makino: On calculating the inverse of separation matrix in frequency-domain blind source separation, ICA'06 (2006) pp. 691–699
52.286 V.C. Soon, L. Tong, Y.F. Huang, R. Liu: A robust method for wideband signal separation, ISCAS'93 (1993) pp. 703–706
52.287 R. Mukai, S. Araki, H. Sawada, S. Makino: Removal of residual cross-talk components in blind source separation using LMS filters, NNSP'02 (2002) pp. 435–444
52.288 H. Saruwatari, S. Kurita, K. Takeda, F. Itakura, K. Shikano: Blind source separation based on subband ICA and beamforming, ICSLP'00, Vol. III (2000) pp. 94–97
52.289 S.Y. Low, S. Nordholm, R. Togneri: Convolutive blind signal separation with post-processing, IEEE Trans. Speech, Audio Process. **12**(5), 539–548 (2004)
52.290 H. Saruwatari, T. Kawamura, T. Nishikawa, A. Lee, K. Shikano: Blind source separation based on a fast-convergence algorithm combining ICA and beamforming, IEEE Trans. Audio, Speech, Lang. Process. **14**(2), 666–678 (2006)
52.291 R. Mukai, H. Sawada, S. Araki, S. Makino: Near-field frequency domain blind source separation for convolutive mixtures, Proc. ICASSP'04, Vol. IV (2004) pp. 49–52
52.292 R.O. Schmidt, R.E. Franks: Multiple source DF signal processing: an experimental system, IEEE Trans. Ant. Prop. **4**(3), 281–290 (1986)
52.293 N. Murata, S. Ikeda: An on-line algorightm for blind source separation on speech signals, International Symposium on Theory and its Applications, Vol. 3 (1998) pp. 923–926
52.294 F. Asano, S. Ikeda: Evaluation and real-time implementation of blind source separation system using time-delayed decorrelation, ICA'00 (2000) pp. 411–416
52.295 S. Ukai, H. Saruwatari, T. Takatani, R. Mukai: Multistage SIMO-model-based blind source sep-

aration combining frequenct-domain ICA and time-domain ICA, Proc. ICASSP'04, Vol. IV (2004) pp. 109–112

52.296 S. Araki, S. Makino, A. Blin, R. Mukai, H. Sawada: Blind separation of more speech than sensors with less distortion by combining sparseness and ICA, IWAENC'03 (2003) pp. 271–274

52.297 R. Mukai, H. Sawada, S. Arakiand, S. Makino: Blind source separation for moving speech signals using blockwise ICA and residual crosstalk subtraction, IEICE Trans. Fundamentals **E87-A**(8), 1941–1948 (2004)

52.298 R. Mukai, H. Sawada, S. Araki, S. Makino: Frequency domain blind source separation for many speech signals, ICA'04 (2004) pp. 461–469

52.299 Y. Mori, H. Saruwatari, T. Takatani, K. Shikano, T. Hiekata, T. Morita: ICA and binary-mask-based blind source separation with small directional microphones, ICA'06 (2006) pp. 649–657

52.300 T. Nishikawa, H. Saruwatari, K. Shikano: Stable learning algorithm for blind separation of temporally correlated signals combining multistage ICA and linear prediction, ICA'03 (2003) pp. 337–342

52.301 T. Takatani, T. Nishikawa, H. Saruwatari, K. Shikano: Blind separation of binaural sound mixtures using SIMO-model-based independent component analysis, Proc. ICASSP'04, Vol. IV (2004) pp. 113–116

52.302 T. Hoya, A.K. Barros, T. Rutkowski, A. Cichocki: Speech extraction based upon a combined subband independent component analysis and neural memory, ICA'03 (2003) pp. 355–360

50. Microphone Arrays

G. W. Elko, J. Meyer

This chapter introduces various types of microphone array beamforming systems and discusses some of the fundamental theory of their operation, design, implementation, and limitations. It is shown that microphone arrays have the ability to offer directional gains that can significantly improve the quality of signal pickup in reverberant and noisy environments.

Hands-free audio communication is now a major feature in mobile communication systems as well as audio and video conferencing systems. One problem that becomes evident to users of these systems is the decrease in communication quality due to the pickup of room reverberation and background noise. In the past, this problem was dealt with by using microphones placed close to the desired talker or source. Although this simple solution has proven to be quite effective, it also has its drawbacks. First, it is not always possible or desirable to place the microphone very close to the talker's mouth. Second, by placing the microphone close to the talker's mouth, one has to deal with rapid level variation as the talker moves his or her mouth relative to the microphone. Third is the negative impact of speech plosives (airflow transients generated by plosive sounds) and forth, microphone structure-borne handling noise has a detrimental effect. Finally, for directional microphone elements, there is also a nearfield proximity effect (where the frequency response of the microphone is modulated by the relative position of the microphone to the mouth). With these issues in mind, it is of interest to investigate other potential solutions. One solution is to use beamforming microphone arrays, which can offer significant directional gain so as to result in similar audio performance to that of closely placed microphones.

50.1 Microphone Array Beamforming

The propagation of acoustic waves in space is a function of both space and time coordinates. In general, the mathematical representation of a propagating acoustic wave is a four-dimensional function: three spatial dimensions and one time variable. Acoustic wave propagation can be modeled by a linearized scalar acoustic wave equation that describes the relationships between the physical quantities of acoustic pressure and density variation of the medium [50.1],

$$\nabla^2 p = \frac{1}{c^2}\frac{\partial^2 p}{\partial t^2}, \tag{50.1}$$

where p is the instantaneous acoustic pressure fluctuation of the sound and c is the propagation speed of sound in the medium. There are a few major underlying assumptions in the derivation of (50.1), but this simplified linear model is adequate for laying the foundation for array beamforming of acoustic signals. The dimen-

sionality of the wave equation can easily be seen from (50.1), where p is a function of three space variables and one time variable. Therefore the scalar acoustic pressure field in space can be represented as $p(\boldsymbol{r}, t)$, where $\boldsymbol{r}$ is the measurement position in the acoustic field and t is the time dependence. The scalar acoustic pressure field must satisfy (50.1) at all points in space. By using the spatial Fourier transform, the acoustic field can also equivalently be represented in the wavevector–frequency domain. This representation has been widely used in the fields of geophysics, structural, underwater and aeroacoustics, and can be quite useful in the analysis and design of microphone array beamformers.

Applying the four-dimensional Fourier transform to the time–space scalar acoustic field yields,

$$P(\boldsymbol{k}, \omega) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} p(\boldsymbol{r}, t)\, \mathrm{e}^{-\mathrm{i}(\omega t - \boldsymbol{k}^{\mathrm{T}} \boldsymbol{r})}\, \mathrm{d}\boldsymbol{r}\, \mathrm{d}t \,, \tag{50.2}$$

where the acoustic pressure P is capitalized to indicate a transform to the frequency domain, $\boldsymbol{k}$ is the wavevector and the superscript 'T' represents the transpose operator. It may seem that this transformation has unnecessarily complicated the description of the scalar acoustic pressure field, but as will be seen later, it has transformed the field description into a form that makes the analysis of acoustic field space–time functions analogous to the field of multidimensional signal processing. Since the Fourier transform is a linear transformation, one can write an inverse relationship,

$$p(\boldsymbol{r}, t) = \frac{1}{(2\pi)^4} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} P(\boldsymbol{k}, \omega)\, \mathrm{e}^{\mathrm{i}(\omega t - \boldsymbol{k}^{\mathrm{T}} \boldsymbol{r})}\, \mathrm{d}\boldsymbol{k}\, \mathrm{d}\omega \,. \tag{50.3}$$

Equation (50.3) directly shows that any acoustic field $p(\boldsymbol{r}, t)$ can be represented as an infinite number of propagating plane waves with appropriate complex weighting. This interpretation can easily be seen from the equation for a single propagating plane wave with frequency ω_0 and wavevector $\boldsymbol{k}_0$,

$$p(\boldsymbol{r}, t) = A_0\, \mathrm{e}^{-\mathrm{i}(\omega_0 t - \boldsymbol{k}_0^{\mathrm{T}} \boldsymbol{r})} \,, \tag{50.4}$$

where A_0 is the plane-wave amplitude. Note that for generality, one can allow A_0 to be complex, but for this discussion it is assumed that A_0 is real. The wavenumber–frequency spectrum of this single propagating plane wave is simply,

$$P(\boldsymbol{k}, \omega) = A_0 \delta(\boldsymbol{k} - \boldsymbol{k}_0) \delta(\omega - \omega_0) \,. \tag{50.5}$$

Equation (50.5) shows the Fourier mapping of infinite continuous functions in both time and space to be a point in the wavevector–frequency space.

Now consider a general weighting function of the space–time pressure distribution with a four-dimensional linear shift-invariant filter having an impulse response $h(\boldsymbol{r}, t)$. Then, the output signal is the convolution of the space–time acoustic pressure signal and the spatial weighting function such that,

$$y(\boldsymbol{r}, t) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(\boldsymbol{r} - \hat{\boldsymbol{r}}, t - \tau) p(\hat{\boldsymbol{r}}, \tau)\, \mathrm{d}\hat{\boldsymbol{r}}\, \mathrm{d}\tau \,. \tag{50.6}$$

Equation (50.6) is a filtering operation that can equivalently be represented in the wavevector–frequency domain as,

$$Y(\boldsymbol{k}, \omega) = H(\boldsymbol{k}, \omega) P(\boldsymbol{k}, \omega) \,. \tag{50.7}$$

Equation (50.6) and (50.7) show the direct analogy between the space–time wavevector–frequency representation of spatial filtering and the well-known results from multidimensional linear systems theory. Thus, by using specific spatial weighting functions, one can filter the acoustic field to investigate the propagating directions, amplitude, and phases of plane waves traveling in space. Filtering the spatiotemporal acoustic scalar pressure field is known in the field of array signal processing as *beamforming*. Thus, if one wants to *look* at plane waves propagating from the direction of the unit vector $\boldsymbol{k}_0 / \|\boldsymbol{k}_0\|$, one would design a filter such that,

$$H(\boldsymbol{k}, \omega) = \delta(\boldsymbol{k} - \boldsymbol{k}_0) G(\omega) \,, \tag{50.8}$$

where $G(\omega)$ is the desired frequency response to waves propagating in the direction of the wavevector $\boldsymbol{k}_0$.

Although it may seem limiting to use plane waves as the underlying basis functions for the representation of a sound-field, it should be noted that other orthogonal representations such as spherical and cylindrical basis functions can themselves be represented as a series of plane waves.

The representation of the wavevector–frequency domain is a natural framework for analysis of the spatial and frequency filtering of beamforming arrays, as will be seen in the following.

In spatial filtering by beamforming, one deals with spatial apertures. An aperture is a region over which energy is received. Where an aperture can either be continuous, as in parabolic dishes, or discretely realized as in microphone arrays with multiple microphones, mixtures of continuous and discrete apertures are possible.

In fact, even discrete microphone arrays are a mixture since no microphone is a perfect point receiver. Although sampled aperture systems must generally perform more signal processing, they offer several advantages over continuous aperture systems. A main advantage in using sampled apertures is that they allow for the possibility of using digital signal processing on the individual array signals. Arrays can be electronically steered, can form multiple simultaneous beams, and can be made adaptive purely by electronic means.

50.1.1 Delay-and-Sum Beamforming

Delay-and-sum beamforming, also known as *classical beamforming*, is one of the simplest and oldest techniques for realizing directional array systems. Although not fundamentally limited in bandwidth, early delay-and-sum arrays were used in *narrowband* operation to focus arrays onto a particular point or direction. Since a time delay for narrowband applications can be accomplished with a unique phase shift for each element, narrowband beamforming is typically implemented with phase shifts and is therefore commonly referred to as *phased-array* beamforming. One typical application for microphone arrays is to pick up speech or acoustic signals that are wideband (the desired signals cover many octaves in bandwidth). Therefore most microphone arrays for speech acquisition are implemented as delay-sum beamformers since they are inherently wideband.

Both continuous and discrete beamformers can be seen as implementations of wavevector–frequency filtering as discussed in the previous section. It should be noted that these two beamformers are not mutually exclusive; one can, for instance, have a discrete set of continuous transducers so that the beamformer is a mix of continuous and discrete sampling. Of special interest here is the spatial and frequency response of a finite array of microphones that sample the acoustic field. A classic delay-sum beamformer uses the sum of weighted and delayed samples from an array of N discretely sampled points of the pressure field. In general, a delay-sum beamformer output y is formed as,

$$y(t) = \sum_{n=1}^{N} w_n s(\boldsymbol{r}_n, t - \tau_n) \,, \tag{50.9}$$

where the weights w_n are real and the time delay τ_n is applied to the measured signals $s(\boldsymbol{r}_n, t)$, at microphone n. If the incident field is a planewave with wavevector $\boldsymbol{k}_0$, amplitude A_0 and frequency ω_0, then,

$$s(\boldsymbol{r}_n, t) = A_0 \, \mathrm{e}^{\mathrm{i}(\omega_0 t - \boldsymbol{k}_0^{\mathrm{T}} \boldsymbol{r}_n)} \,, \tag{50.10}$$

and

$$y(t) = A_0 \, \mathrm{e}^{\mathrm{i}\omega_0 t} \sum_{n=1}^{N} w_n \, \mathrm{e}^{-\mathrm{i}(\omega_0 \tau_n + \boldsymbol{k}_0^{\mathrm{T}} \boldsymbol{r}_n)} \,. \tag{50.11}$$

The real weights w_n scale the signals measured at the positions $\boldsymbol{r}_n$. Thus, from (50.9), the origin of the term *delay-and-sum* can be seen as simply a description of how the classical delay-sum beamformer is realized.

For a causal delay-sum beamformer, the exponent in (50.11) must be less than or equal to zero for all n. Typically, one element position is selected as the spatial origin. This reference element position therefore defines the vectors $\boldsymbol{r}_n$. Since the array can be steered to any direction in space and one can select any position to define $\boldsymbol{r}_n$, an additional delay may be required in order to maintain causality. (Recall that $\boldsymbol{k}_0^{\mathrm{T}} \boldsymbol{r}_n$ can be either positive or negative depending on the direction of the incident wave relative to the array.) If an end element is selected as the spatial origin, then the causal delay is equal to the maximum time that an acoustic wave takes to transit the array. Smaller causal delays are also possible depending on which microphone position in the array is chosen as the spatial origin reference and the maximum desired steering angle.

If we set the causal time delay equal to T_0, then the output for an incident plane wave with angular frequency ω_0 and wavevector $\boldsymbol{k}_0$ can be written as,

$$y(t) = A_0 \, \mathrm{e}^{\mathrm{i}\omega_0 t} \sum_{n=1}^{N} w_n \, \mathrm{e}^{-\mathrm{i}\left[\omega_0 (T_0 + \tau_n) + \boldsymbol{k}_0^{\mathrm{T}} \boldsymbol{r}_n\right]} . \tag{50.12}$$

The maximum plane-wave response corresponds to the direction where the delays τ_n compensate for the propagation delay $\boldsymbol{k}_0^{\mathrm{T}} \boldsymbol{r}_n$ of a plane wave propagating with wavevector $\boldsymbol{k}_0$. This is done by setting the values of τ_n such that these delays offset the time lead (or lag) of a desired direction plane wave propagating over the array. Thus, setting the delays to steer the array to the direction of $\boldsymbol{k}_0$, means that the last two terms in the exponential in (50.12) cancel out. The delayed and summed output $y(t)$ is now such that all microphone signals propagating from the desired direction are added in phase. This direction corresponds to a maximum amplitude output for any selection of τ_n. Plane waves propagating from directions other than $\boldsymbol{k}_0$ will result in the addition of position-dependent phase variations, and as a result, the output amplitude will be smaller due to destructive interference.

To summarize, classical delay-and-sum beamforming uses delays between each array element that compensate for differences in the propagation delay of

the desired signal direction across the array. Signals originating from a desired direction (or location if the source is in the near field) are summed in phase, while other signals undergo destructive interference. By adjusting the weights of the delay-sum beamformer, the shape of the beam and sidelobes as well as the position of the nulls of an array can be controlled.

Uniformly Spaced Linear Arrays

While array geometries are in principle arbitrary, certain configurations are especially useful and amenable to mathematical analysis. For example, consider a simple array geometry of uniformly spaced collinear elements as shown in Fig. 50.1. Assume that an odd number of elements, numbered from $-N$ to N, are positioned on the x-axis such that element n has coordinate nd, and d is the interelement spacing. Thus, *endfire* directions lie on the x-axis, while *broadside* directions lie in the y–z plane. Much as linear time-domain systems are analyzed by their response to complex exponential time functions, an array response can be considered in terms of its response to complex exponential plane waves (as shown above).

For a collinear delay-and-sum beamformer steered towards a direction making an angle θ_0 measured clockwise from the y-axis, the relative time delays τ_n are,

$$\tau_n = n\frac{d}{c}\sin\theta_0, \tag{50.13}$$

where θ_0 is the *steering* direction, and represents the direction from which plane waves are received with maximum response.

Now, consider the unit-amplitude plane wave of frequency ω_0, arriving from a direction θ (not necessarily equal to θ_0). This wave is represented by the expression,

$$p(\boldsymbol{r}, t) = \mathrm{e}^{\mathrm{i}(\omega_0 t - \boldsymbol{k}_0^{\mathrm{T}}\boldsymbol{r})}\,. \tag{50.14}$$

Note that it has been assumed that the microphones are located along a line so the vector $\boldsymbol{r}$ points along the axis of the array. Since the array is axisymmetric about the axis of the array, the components of the wavevector can be assumed, without loss of generality, to lie in the x–y plane. With this assumption, one only needs to investigate the array response in the plane. The wavevector can be written into its three orthogonal components as,

$$\boldsymbol{k}_0 = -\frac{\omega_0}{c}\begin{pmatrix}\sin\theta\\ \cos\theta\\ 0\end{pmatrix}\,. \tag{50.15}$$

Substituting the wave field for a planewave and time delay expressions into the beamformer output expression yields,

$$y(t) = \mathrm{e}^{\mathrm{i}\omega_0 t}\sum_{n=-N}^{N} w_n\,\mathrm{e}^{\mathrm{i}\frac{\omega_0}{c}nd(\sin\theta-\sin\theta_0)}\,. \tag{50.16}$$

This expression can be simplified by removing the time-varying $\mathrm{e}^{\mathrm{i}\omega_0 t}$ term and substituting $u = \sin\theta$ and $u_0 = \sin\theta_0$ rather than by the angle itself. This convention has several advantages beyond mere compactness. First, a change of steering direction amounts to a single linear translation in u-space. Thus, the shape of the beam measured in the u-domain does not change with array steering angle, while beams measured in θ space change as a function of the steering direction. Second, one can logically divide the description of the array function into two regions: a *visible* and *invisible* region [50.2]. The visible region corresponds to values of u that have real angles in space and corresponds mathematically to $|u| \le 1$. Values of $|u| > 1$ correspond to angles that are not physical and is therefore referred to as the invisible region.

Making the substitutions for $\sin\theta$ and $\sin\theta_0$ yields an expression for the *array response*,

$$H(u, \omega) = \sum_{n=-N}^{N} w_n\,\mathrm{e}^{\mathrm{i}\frac{\omega}{c}nd(u-u_0)}. \tag{50.17}$$

Note that the description of the array response given in (50.17) has been written as a function of the two independent variables u and ω instead of the variables $\boldsymbol{k}$, t, and ω. This change of variables allows one to better visualize the response of the array in terms of angle and frequency.

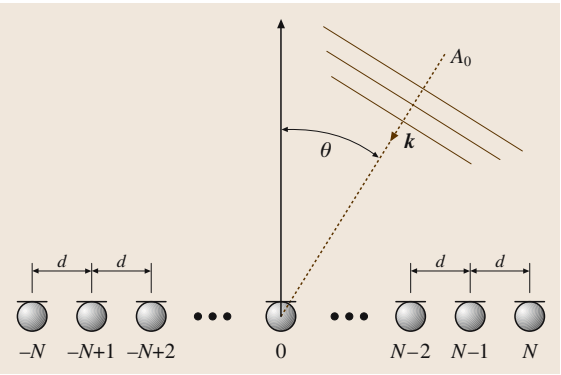


Fig. 50.1 Schematic of uniformly spaced linear array containing $2N+1$ microphones spaced by d

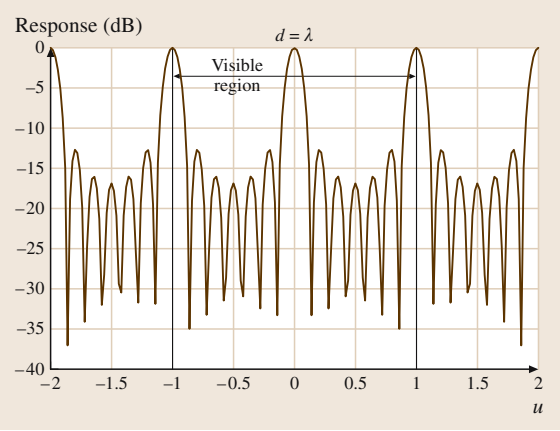


Fig. 50.2 Response of a seven-element uniformly spaced and weighted array as a function of the dimensionless variable u for element spacing equal to the wavelength of the incident acoustic signal

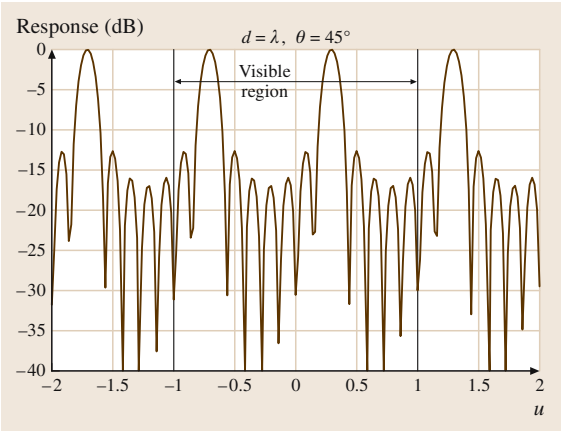


Fig. 50.3 u-response of a seven-element uniformly spaced and weighted array steered 45° as a function of the dimensionless variable u for element spacing equal to the wavelength of sound

The array response as expressed in (50.17) is a function in a two-dimensional u–ω space. For any real constant K the following identity holds,

$$H\left[\left(\frac{u}{K}+u_0\right), K\omega\right] = H(u+u_0, \omega)\,. \tag{50.18}$$

From (50.18), it can be seen that the spatial extent (beamwidth) of the array becomes commensurately smaller as the frequency increases directly proportionally to the frequency ω. Thus if the incident wave frequency is doubled, the beamwidth of the array is cut in half and vice versa. Another important aspect of the array function is that it is periodic in u for any given ω,

$$H(u, \omega) = H\left(u + \frac{2\pi mc}{\omega d}, \omega\right)\,, \tag{50.19}$$

where m is an integer. Thus, the main lobe has an infinite set of identical copies, which are called *grating lobes*. Since the *visible region* [50.2] is defined for values of $-1 \le u \le 1$, grating lobes may or may not be seen in the array response. As can be seen in (50.19), the appearance of grating lobes is a function of both microphone spacing and incident frequency.

Grating lobes are visible when another lobe appears at $|u| \le 1$. When fully visible, a grating lobe is equal in amplitude to the main lobe of the array. This phenomena is called spatial aliasing and is directly analogous to time aliasing in sampled time systems. As either the element spacing or the frequency increases, grating lobes move in an angular direction towards the main lobe and can be thought of as either a dilatation or contraction of the array function as defined in (50.19). As the spacing or frequency increases, new grating lobes become visible, appearing first along the endfire direction. If the array is not steered, grating lobes occur when the first periodic replication $(2\pi c)/\omega$ falls into the visible region $|u| \le 1$. In terms of the spacing d, grating lobes occur when

$$d \ge \frac{2\pi c}{\omega} = \lambda\,, \tag{50.20}$$

where λ is the incident sound wavelength. As shown above, the spatiofrequency response $H(u, \omega)$ is periodic, and thus *spatial aliasing* for spatiotemporal signals will occur before the grating lobe appears at $|u| = 1$. Thus, to preclude spatial aliasing while having any steering angle $|u_0| \le 1$, the grating lobe must be be out of visual space, $|u| + |u_0| \ge 2$. With this constraint,

$$d < \frac{\pi c}{\omega} = \frac{\pi c}{2\pi f} = \frac{\lambda}{2}\,. \tag{50.21}$$

Equations (50.20) and (50.21) define the *spatial sampling* requirement, which is analogous to the Nyquist sampling rate for sampled time-domain signals. If the array is steered to endfire which is the maximum steering angle in visible space, grating lobes can be avoided if adjacent lobes are separated by a distance greater than 2 in u-space (the same restriction as for no spatial aliasing for an unsteered array). It should also be noted that the lowest frequency at the onset of a grating lobe can be increased if the array is composed of directional elements that have low sensitivity in the region of $|u| \approx 1$.

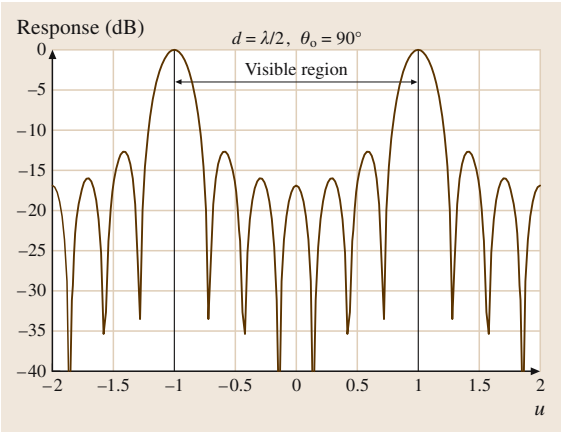


Fig. 50.4 u-response of a seven-element uniformly spaced and weighed array steered 90° as a function of the dimensionless variable u for element spacing equal to half the wavelength of sound

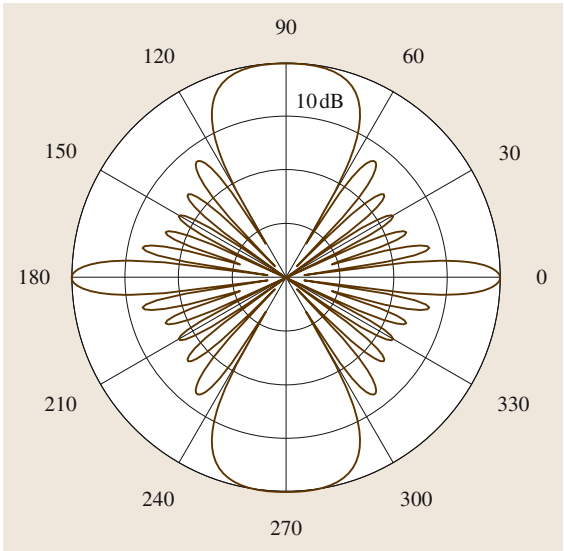


Fig. 50.5 Polar directivity pattern for a seven-element uniformly spaced and weighted unsteered array when $d = \lambda$. Note the large grating lobe at $\theta = 90°$. Also note that the directional response is axisymmetric around the $\theta = 0°$ axis

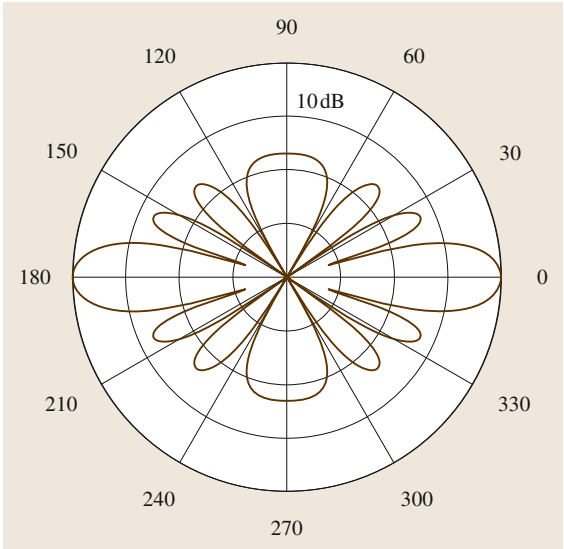


Fig. 50.6 Polar directivity pattern for a seven-element uniformly spaced and weighted unsteered array for $d = \lambda/2$. Note that compared to Fig. 50.5, there is no grating lobe in the $\theta = 90°$ direction

Uniformly Sampled and Weighted Linear Arrays

To better understand spatial aliasing and how $H(u, \omega)$ changes with increased spacing or frequency, as well as steering angle, it is instructive to examine a uniformly weighted linear array. Figure 50.2 shows the response of a uniformly weighted array as a function of the variable u for the case $d = \lambda$. The visible region is denoted on the figure as the region between the two vertical red lines. One can clearly see the grating lobes beginning to move into the visible region in the figure at $u = \pm 1$. This behavior was predicted by (50.20) for an unsteered array. Figure 50.3 shows how the response is rotated in u-space when the array is steered 45° from the unsteered direction (broadside). One can now clearly see that the grating lobe is fully within the visible region. For comparison purposes, Fig. 50.4 shows the directional response for a 90° steered array for the case of $d = \lambda/2$. Equation (50.21) predicts that for this case, a grating lobe should appear in the opposite direction (along the axis of the array but in the opposite of the steered direction). Figure 50.4 confirms that this is indeed the case as one can see the grating lobe at 270° when the array is steered to 90° (endfire).

Figures 50.5 and 50.6 show directivity patterns when $d = \lambda$ and $d = \lambda/2$ for a seven-element uniformly spaced and weighted unsteered array. It is not coincidental that there are six zeros (nulls) in the response since as can be seen in (50.17), a seven-element array response can be written as a seventh-order polynomial. As the frequency or spacing changes, the spatial response either contracts or dilates, so at lower frequencies or smaller element spacing, some or all of these nulls might not be contained within the visible region $|u| \leq 1$. One way to visualize

this effect is to plot the angular response of the array as a function of frequency.

Figure 50.7 shows a surface plot of the magnitude of the response of a seven-element array with 8 cm spacing versus angle and frequency. The colors indicate a logarithmic range limited to 40 dB. The narrowing of the main lobe as frequency increases is clearly evident in the figure. It can also be seen that the array is essentially omnidirectional below 500 Hz and that the first spatial aliasing lobe appears at around 4 kHz while a second grating lobe appears at around 8 kHz. These two frequencies correspond to sound with wavelengths of 8 and 16 cm, respectively. Thus grating lobes appear for the unsteered array at frequencies where the element spacing is an integer multiple of the incident sound wavelength, as predicted by (50.20). Six distinct zeros can be seen in the response between the grating lobes in Fig. 50.7 and it can also be seen that they gradually disappear from the visible region below 4 kHz. No nulls occur in the visible region below 500 Hz.

The angular position of the first nulls on either side of the main lobe can be computed for an unsteered uniformly weighted array by noting that (50.17) can be rewritten

$$H(u,\omega) = \frac{\sin(\hat{N}\omega u d/2c)}{\sin(\omega u d/2c)}\,, \tag{50.22}$$

where $\hat{N} = 2N+1$ is the total number of elements, The first null occurs when the argument of the numerator of (50.22) is $\pm\pi$. The beamwidth, $\mathrm{BW}_{\mathrm{null}}$ is twice the angle between the main lobe and the first null

$$\begin{aligned}\mathrm{BW}_{\mathrm{null}} &= 2\sin^{-1}\left(\frac{2\pi c}{\hat{N}\omega d}\right)\\ &= 2\sin^{-1}(\lambda/L)\,,\end{aligned} \tag{50.23}$$

where λ is the wavelength of the incident sound and $L = \hat{N}d$ is close to the array length $[(\hat{N}-1)d]$. (Note that other definitions for beamwidth are also used, such as the angles where the response falls to half the mainlobe level.) If the array length L is much larger than the incident wavelength, then one can use the small-angle approximation and obtain,

$$\mathrm{BW}_{\mathrm{null}} \approx \frac{2\lambda}{L}\,. \tag{50.24}$$

Equations (50.23) and (50.24) show that the beamwidth is proportional to the wavelength and inversely proportional to the length of the array.

Although the above analysis is only for uniformly weighted and spaced linear arrays, the general qualitative relationships between the array length, frequency, and beamwidth are similar for weighted and nonuniformly sampled arrays.

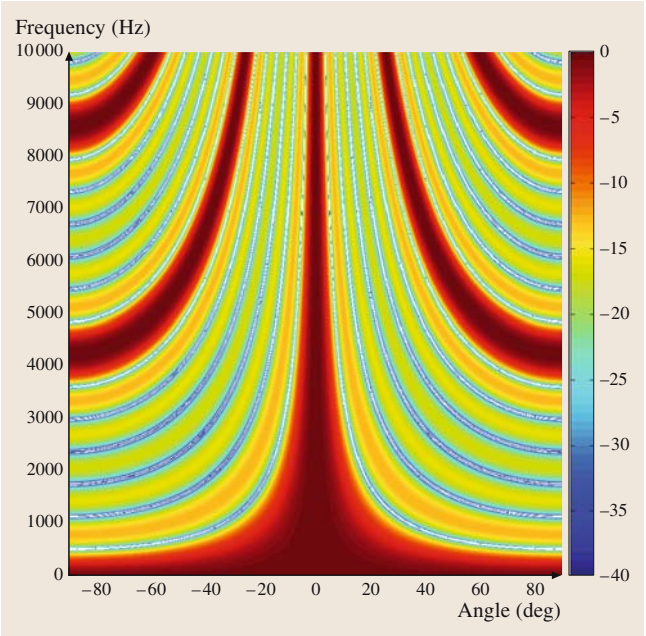


Fig. 50.7 Response of a seven-element uniformly spaced and weighted array for an element spacing of 8 cm

Analogy with FIR Filter Design

The discussion thus far expresses the beamformer behavior in terms of the array weights w_n. The opposite process, finding the weights for a given desired behavior, is also extremely important. This process is now shown to be related to finite impulse response (FIR) filter design.

Consider the Fourier transform of the array weights,

$$X(\mathrm{e}^{\mathrm{i}\omega_{\mathrm{t}}}) = \sum_{n=-N}^{N} w_n\,\mathrm{e}^{-\mathrm{i}\omega_{\mathrm{t}}n}\,, \tag{50.25}$$

where $\omega_{\mathrm{t}} = (u-u_0)\omega_0 d/c$. Comparing this with (50.17) yields the identity

$$H\left(u_0 + \frac{\omega_{\mathrm{t}}c}{\omega_0 d}, \omega_0\right) = X(\mathrm{e}^{\mathrm{i}\omega_{\mathrm{t}}})\,, \tag{50.26}$$

for any ω_0. From (50.26), it can be seen that finding the weights w_n for a given $H(u,\omega_0)$ is equivalent to an FIR filter design problem for a given desired frequency response $X(\mathrm{e}^{\mathrm{i}\omega_{\mathrm{t}}})$. Thus, like FIR filter design, array beamformer design is based on finding the appropriate coefficients of a finite-order polynomial that match a design goal in some desired way.

Classical delay-and-sum beamformer element weights are constants and therefore not a function of frequency. Since the spatial response of a delay-and-sum beamformer is a function of frequency, where the beamwidth of the array is commensurately smaller as the frequency increases, it is of interest to find a more-flexible design method that allows control of the beam pattern as a function of frequency. An obvious modification is to generalize the delay-and-sum beamformer by replacing the set of scalar weightings by a set of general filters. For obvious reasons, this generalized beamformer architecture is widely known as a filter-and-sum beamformer.

50.1.2 Filter-and-Sum Beamforming

As discussed in the previous section, it is desirable to enable the control of a beamformer design over a wide frequency range. The analysis of the delay-and-sum beamformer can be extended to a discrete-time filter-and-sum beamformer. Thus, without a loss in generality, the elemental filters are assumed to be FIR filters of length M. The beamformer output sequence, for integer κ, is

$$y[\kappa] = \sum_{n=-N}^{N} \sum_{m=0}^{M-1} p\left[\boldsymbol{r}_n, (\kappa - m)\mathrm{T}\right] \; h_n[m] \,, \quad (50.27)$$

where κ is the time sample, $h_n[m]$ is the impulse response of the filter associated with the n–th element (i. e., the n-th elemental filter), the number of microphones is $2N+1$, T is the sampling period, and $p(\boldsymbol{r}, t)$ is the incident wave field. As before, we consider the complex exponential plane wave,

$$p(\boldsymbol{r}, \kappa\mathrm{T}) = \mathrm{e}^{\mathrm{i}(\omega_0 \kappa \mathrm{T} - \boldsymbol{k}_0^{\mathrm{T}} \boldsymbol{r})} \,, \quad (50.28)$$

for some frequency ω_0 and wavevector $\boldsymbol{k}_0$. A uniformly spaced beamformer with spacing d has an output

$$y[\kappa] = \mathrm{e}^{\mathrm{i}\omega_0 \kappa \mathrm{T}} \sum_{n=-N}^{N} \sum_{m=0}^{M-1} h_n[m] \mathrm{e}^{\mathrm{i}(-\omega_0 m \mathrm{T} + \frac{\omega_0}{c} u_0 n d)} \,, \quad (50.29)$$

where $u_0 = \sin\theta_0$ as before.

Once again, the time-dependent portion is omitted to give the array response,

$$H(u, \omega) = \sum_{n=-N}^{N} \sum_{m=0}^{M-1} h_n[m] \mathrm{e}^{-\mathrm{i}(\omega m \mathrm{T} - \frac{\omega}{c} u n d)} \,. \quad (50.30)$$

From (50.30) it can now be seen that a filter-sum beamformer design problem is equivalent to a two-dimensional FIR filter design problem.

Two-Dimensional Filter Design

Just as delay-and-sum beamformer design is related to one-dimensional (1-D) FIR filter design, the filter-and-sum beamformer design is related to two-dimensional (2-D) FIR filter design that can be seen by considering an $N \times M$ function of 2-D Dirac delta functions, weighted according to the elemental filter coefficients,

$$f(x, y) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \delta(x - n, y - m) h_n[m] \,. \quad (50.31)$$

Now, the discrete 2-D Fourier transform of this function is:

$$X(\omega_1, \omega_2) = \sum_{n=-N}^{N} \sum_{m=0}^{M-1} h_n[m] \mathrm{e}^{-\mathrm{i}(\omega_1 n + \omega_2 m)} \,. \quad (50.32)$$

Note that this function is periodic with period 2π in both the ω_1 and ω_2 dimensions. Combining this with (50.30) yields

$$H\left(\frac{\omega_1 c \mathrm{T}}{\omega_2 d}, \frac{\omega_2}{\mathrm{T}}\right) = X(\omega_1, \omega_2) \,. \quad (50.33)$$

Equation (50.33) implies that a filter-and-sum beamformer can be conceptualized as a 2-D filter whose Fourier transform is closely related to the desired array response. Note that $X(\omega_1, \omega_2)$ is exactly the familiar wavenumber–frequency response [50.3], after converting the wavenumber and frequency variables $k_x = \|\boldsymbol{k}\| \sin\theta$, ω to the quantities $\omega_1 = k_x d$, $\omega_2 = \omega \mathrm{T}$, which have units of radians.

50.1.3 Arrays with Directional Elements

A microphone element frequency response can be effectively corrected with a single inverse filter placed at the output of the beamformer. Microphone elements with varying directivity patterns cannot generally be compensated for, but one can mathematically model their behavior using the *pattern multiplication* theorem [50.2]. This principle states that the total far-field response can be obtained from the far-field response of an equivalent array constructed using omnidirectional

elements, multiplied by the far-field response of the individual microphones. Thus, the total array response is the product of the two array responses,

$$H(u,\omega) = H_{\mathrm{m}}(u,\omega)H_{\mathrm{a}}(u,\omega)\,,$$

where $H_{\mathrm{m}}(u,\omega)$ is the far-field response of a single microphone element, and $H_{\mathrm{a}}(u,\omega)$ is the far-field response of the equivalent array composed of omnidirectional elements. (Note that all of the directional elements must be identical and oriented in the same manner).

50.2 Constant-Beamwidth Microphone Array System

Due to hardware limitations, early digital beamforming microphone arrays were implemented as delay-sum beamformers [50.4]. As discussed previously in (50.19), a delay-and-sum beamformer has a decreasing beamwidth as the frequency increases. In fact, (50.23) showed that these functions are inversely related. Narrowing of the beamwidth as frequency increases can result in an undesired low-pass-colored output from an array if it is placed into a typical room with reverberation. This is due to the larger beamwidth at lower frequencies that allows more input power from a reverberant acoustic field. To minimize the coloration of a desired wide-band signal, it is desirable to consider beamformers whose directivity is constant over frequency.

By examining the results given in (50.23) and (50.24), an obvious solution towards attaining a constant-beamwidth beamformer design would be to design a filter-sum beamformer where the ratio of the acoustic wavelength to effective array length was held constant or nearly constant.

One proposed solution to this problem was to split the array into subarrays that covered different

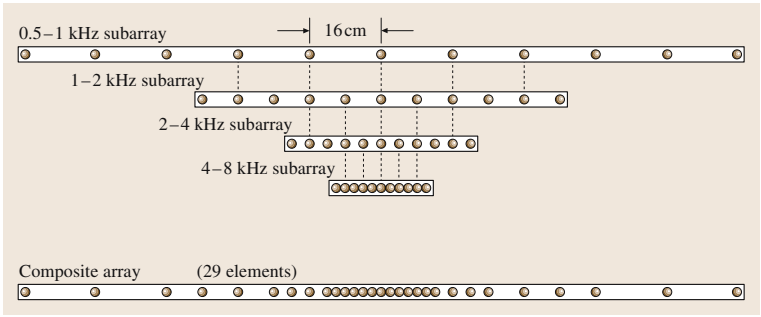


Fig. 50.8 Diagram of nested four-subarray microphone array

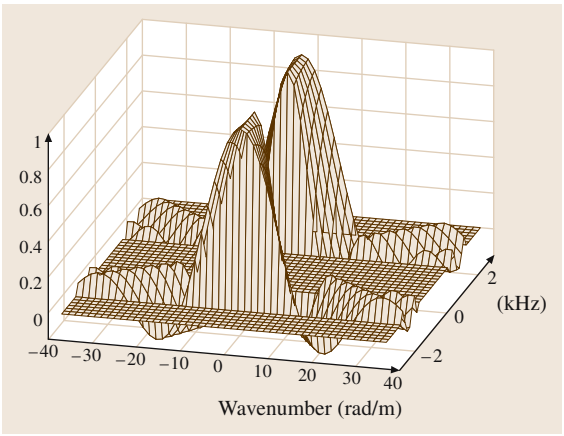


Fig. 50.9 Frequency–wavenumber design for a constant-beamwidth beamformer over one subarray

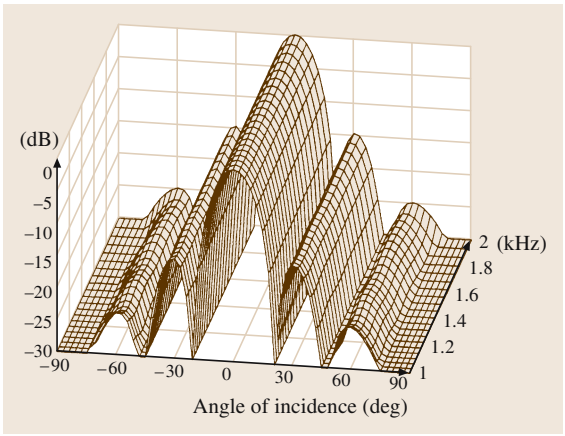


Fig. 50.10 Array response over one octave for a constant directivity array using a fan filter as shown in Fig. 50.9

frequency bands [50.4]. Some early microphone arrays were hybrid digital systems composed of three harmonically-nested subarrays covering a bandwidth from 300 Hz to 4 kHz [50.4, 5], but were later modified to include four harmonically nested subarrays (200–8 kHz). A four-band harmonically nested subarray is shown diagrammatically in Fig. 50.8. One interesting observation that can be made here is that, by harmonically nesting the subarrays, many elements are commonly shared in the array. Thus, the total number of array elements can be significantly reduced.

Further refinements were made on the multi-octave arrays to allow experimentation with a filter-sum configuration [50.6, 7]. These were developed to allow investigation into array algorithms for constant-beamwidth and dynamic control of the array beamwidth [50.7, 8]. Figure 50.9 shows the frequency–wavenumber response of the *fan filter* used to keep the beamwidth constant over the bandwidth of one octave subarray. A basic detail to notice here is that, as the frequency rises, the wavenumber bandwidth commensurately increases so that the beamwidth remains essentially constant. Figure 50.10 shows the array response over the 1–2 kHz octave when using the fan-filter design as shown in Fig. 50.9. It can be seen that the directivity is effectively constant over the whole octave. Similar fan filters are used for all octaves of the microphone array resulting in a constant beamwidth over most of the frequency range [50.6, 7].

The fan-filter design is equivalent to constant directivity beamformers based on the use of different low-pass filters on each element where the low-pass cutoff frequency is proportional the position of the element relative to the center of the array [50.9]. Intuitively the low-pass design acts to shorten the length of the array inversely proportional to frequency, thereby exactly counteracting the inverse relationship between array beamwidth and frequency.

50.3 Constrained Optimization of the Directional Gain

Directional gain is a common measure used in the design and analysis of microphone arrays. Optimization of the directional gain may not yield practical or realizable array designs, and therefore it is necessary to constrain the optimization to attain a useful design.

The directivity factor metric is typically used in quantifying the change in the acoustic signal-to-noise ratio (SNR) of a beamforming array in an isotropic noise field. The directivity factor (Q) expresses the ratio of sound energy received from the steered direction to the average energy received from all directions. Mathematically, the definition is,

$$Q(r,\theta_0,\phi_0)=\frac{4\pi\,|y(r,\theta_0,\phi_0)|^2}{\int_0^{2\pi}\int_0^{\pi}|y(r,\theta,\phi)|^2\sin\theta\,\mathrm{d}\theta\,\mathrm{d}\phi}\,, \tag{50.34}$$

where $y(r,\theta,\phi)$ is the response to a wave arriving from spherical coordinate directions (r,θ,ϕ), and (θ_0,ϕ_0) the steering directions. It can be shown that this function can be maximized for any array geometry by solving for the maximum of the ratio of two Hermitian forms [50.10]. Although it is not always the goal to maximize the directivity factor, this quantity is typically used to quantify beamforming acoustic SNR gains for far-field sources in isotropic noise fields. It is standard engineering practice to express the directivity factor in decibels, and this form is called the directivity index. The directivity index (DI) is defined as

$$\mathrm{DI}(r,\theta_0,\phi_0)=10\log\left[Q(r,\theta_0,\phi_0)\right]. \tag{50.35}$$

Optimization of the directional gain at frequencies where the array size is smaller than the acoustic wavelength results in array designs that are referred to as superdirectional arrays. Superdirectional arrays obtain directivity by forming the differences between subarray beampatterns. Superdirectional arrays have element weighting functions that oscillate in and out of phase between elements and can have directional indices (DI) potentially twice as high as classical delay-sum beamformers [$20\log(N)$ versus $10\log(N)$] [50.11]. Pattern-differencing beamformers result in a reduced acoustic signal output relative to microphone self-noise and electronics. Thus the array output SNR that can be significantly compromised. This problem has been known in the past and many authors have written on the subject [50.12]. The earliest publication containing a solution that constrains the gain to deal with the problem was a paper by *Gilbert* and *Morgan* [50.13]. However, it was not until a publication by *Cox* et al. [50.14] that a practical design methodology emerged. The following development follows that of Cox et al.

Consider a linear array of N sensors with desired signal and noise spatial cross-correlation matrices de-

noted by $\boldsymbol{P}$ and $\boldsymbol{Q}$, respectively. If one assumes that the signal and noise signals are stationary and independent processes, then the spatial cross-correlation matrix for the array can be written as

$$\boldsymbol{R}(\omega) = \sigma_{\mathrm{s}}^2 \boldsymbol{P}(\omega) + \sigma_{\mathrm{n}}^2 \boldsymbol{Q}(\omega)\,, \tag{50.36}$$

where the frequency domain is denoted by the explicit use of the temporal frequency ω. Thus the input signal-to-noise ratio is simply $\sigma_{\mathrm{s}}^2/\sigma_{\mathrm{n}}^2$. The output power spectrum of the beamformer that combines each microphone signal by a weight $w(\omega)$ can be written

$$Y = \boldsymbol{w}^{\mathrm{H}} \boldsymbol{R} \boldsymbol{w}, \tag{50.37}$$

where $\boldsymbol{w}$ is the complex weight vector containing the beamformer weights for each microphone element at frequency ω_0, and $^{\mathrm{H}}$ denotes the complex conjugate transpose. Also note that the explicit frequency dependence has been omitted for compactness of the expression. The array gain (equivalent to the directivity index for an isotropic noise field) can now be defined as the relative gain in SNR at the beamformer output, and can be written as,

$$G = \frac{\boldsymbol{w}^{\mathrm{H}} \boldsymbol{P} \boldsymbol{w}}{\boldsymbol{w}^{\mathrm{H}} \boldsymbol{Q} \boldsymbol{w}}\,, \tag{50.38}$$

Note that in general, $\boldsymbol{Q}$ can also contain components of sensor self-noise.

In order to quantify the gain in SNR through a beamformer, *Cox* [50.14] proposed a measure called *white-noise gain* (WNG), which simplifies the above equations by assuming that the noise matrix can be assumed as an identity matrix. An expression of the WNG is then simply

$$G_{\mathrm{w}} = \frac{\boldsymbol{w}^{\mathrm{H}} \boldsymbol{P} \boldsymbol{w}}{\boldsymbol{w}^{\mathrm{H}} \boldsymbol{w}} \tag{50.39}$$

From (50.39), it can be seen that lower values of WNG imply a higher relative sensitivity to white-noise signals in each microphone relative to the sensitivity of acoustic signals. Therefore, the term WNG is best understood when interpreted as the gain of the array *over* white noise.

A constrained design can be obtained by the use of Lagrange multipliers such that

$$\begin{aligned} &\arg\min_{\boldsymbol{w}} \left\{ \frac{1}{G} + \epsilon \frac{1}{G_{\mathrm{w}}} \right\} \\ &= \arg\min_{\boldsymbol{w}} \left\{ \frac{\boldsymbol{w}^{\mathrm{H}} [\boldsymbol{Q} + \epsilon \boldsymbol{I}] \boldsymbol{w}}{\boldsymbol{w}^{\mathrm{H}} \boldsymbol{P} \boldsymbol{w}} \right\}\,. \end{aligned} \tag{50.40}$$

Equation (50.40) is a generalized eigenvalue problem and the constrained optimum weight can also be found [50.10] as the eigenvector corresponding to the largest eigenvalue of the middle term in the numerator in (50.40). For a single desired source propagating from a known direction θ relative to the array, one can obtain the optimum weight as,

$$\boldsymbol{w}_{\mathrm{opt}} = [\boldsymbol{Q} + \epsilon \boldsymbol{I}]^{-1} \boldsymbol{s}^*(\theta), \tag{50.41}$$

where $\boldsymbol{s}^*$ is the complex-conjugate signal across the array generated by the desired signal at angle θ. Equation (50.41) can be seen to be the constrained optimum beamformer that operates by the process of *whiten then match*. The *whitening* is done by the inverse of the term in the brackets where areas of higher spatial noise are commensurately attenuated by the beamformer. The second term within the brackets is a regularization of the inverse of the noise correlation matrix and acts as a constraint on the inverse. Matching to waves propagating from the desired direction is accomplished by the steering vector $\boldsymbol{s}^*$ that steers the array in the direction of the desired sound source.

By adjusting the Lagrange multiplier between the values of 0 and ∞, one can obtain a monotonic change in directional gain from an unconstrained array for $\epsilon = 0$ to that of a uniformly weighted array ($\epsilon \approx \infty$).

50.4 Differential Microphone Arrays

Differential microphone arrays are part of a regime of arrays that are of general interest due to their small size relative to the acoustic wavelength. For a given number of microphone elements in an array, differential arrays theoretically have the potential to attain maximum directional gain [50.11]. However, as will be shown later in this section, there are some significant caveats in realizing high directional gain with small differential arrays.

Earlier discussion on delay and filter-sum beamformers showed that these arrays operate as FIR spatial low-pass filters. As a result, directional gain is obtained when the array length is large or on the order of the incident sound wavelength. On the contrary, differential arrays operate as spatial high-pass filters, and as such, can be developed and analyzed differently than the delay and filter-sum arrays. (To be precise, a differential array does fall into the general filter-sum beamformer

category. However, since the analysis of differential array operation can be developed more compactly, it will be assumed that these arrays are different.)

Differential array output can be understood as a finite-difference approximation to the sum of spatial derivatives of the scalar acoustic pressure field. The term *first-order* differential array applies to any array whose response is proportional to the combination of two components: a zeroth-order (acoustic pressure) signal and another proportional to the first-order spatial derivative of a scalar acoustic pressure field. Similarly, the term n-th-*order* differential array is used for arrays that have a response proportional to a linear combination of signal derived from spatial derivatives up to, and including order n.

Before discussing various implementations of n-th-order finite-difference arrays, expressions are developed for the n-th-order spatial acoustic pressure derivative in a direction $\boldsymbol{r}$. Since realizable differential arrays are approximations to acoustic pressure differentials, equations for general order differentials provide significant insight into the operation of these arrays.

The acoustic pressure field for a propagating acoustic plane wave can be written

$$p(\boldsymbol{r}, t) = A_0 \, \mathrm{e}^{\mathrm{i}(\omega_0 t - \boldsymbol{k}_0^{\mathrm{T}} \boldsymbol{r})} = A_0 \, \mathrm{e}^{\mathrm{i}(\omega_0 t - k_0 r \cos\phi)} \,, \tag{50.42}$$

where A_0 is the plane-wave amplitude. The angle ϕ is the angle between the position vector $\boldsymbol{r}$ and the wavevector $\boldsymbol{k}_0$ (note that the angle ϕ here is not the same as the standard spherical coordinate angle used to describe the angle in the x–y plane). Dropping the time dependence and taking the n-th-order spatial derivative along the direction of the position vector $\boldsymbol{r}$ yields:

$$\frac{\mathrm{d}^n}{\mathrm{d}r^n} p(k, r) = A_0 (-\mathrm{i}k \cos\phi)^n \, \mathrm{e}^{-\mathrm{i}kr\cos\phi} \,. \tag{50.43}$$

The plane-wave solution is valid for the response to sources that are *far* from the microphone array. The term *far* implies that the distance between the source and receiver is many times the square of the relevant dimension divided by the acoustic wavelength where the relevant dimension is the greater of the source or receiver dimension. Using (50.43) one can interpret that the n-th-order differential has a bidirectional pattern component with the shape of $(\cos\phi)^n$. It can also be seen that the frequency response of a differential microphone is high-pass with a slope of $6n$ dB per octave. If the far-field assumption is relaxed, the response of the differential system to a point source located at the coordinate origin is

$$p(k, r) = A_0 \frac{\mathrm{e}^{-\mathrm{i}(kr\cos\phi)}}{r} \,, \quad r > 0. \tag{50.44}$$

The n-th-order spatial derivative in the radial direction r is

$$\frac{\mathrm{d}^n}{\mathrm{d}r^n} p(k, r, \phi) = A_0 \frac{n!}{r^{n+1}} \, \mathrm{e}^{-\mathrm{i}kr\cos\phi} (-1)^n \sum_{m=0}^{n} \frac{(\mathrm{i}kr\cos\phi)^m}{m!} \,. \tag{50.45}$$

As can be seen in (50.45), a fundamental property for differential arrays is that the general n-th-order array response is a weighted sum of bidirectional terms of the form $\cos(\phi)^n$.

Implicit in the development of differential arrays is the inherent assumption that the true differential can be approximated by discrete microphones that are used to estimate the pressure differentials by finite differences. For reasonable estimation of the differential by finite differences of the acoustic pressure, the microphones must be placed such that the element spacing is much less than the acoustic wavelength over the desired frequency range. With the small-spacing requirements, one can see both the main advantage and disadvantage of differential arrays: they can attain high directional gain with small size, but also have commensurately small WNG values, indicating a high sensitivity to self-noise as well as microphone amplitude and phase mismatch.

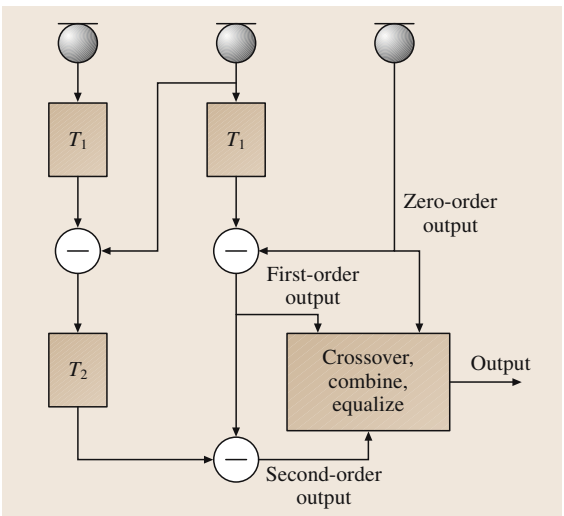


Fig. 50.11 Construction of a generalized second-order differential array as first-order differential combinations

An n-th-order differential array can be written as a sum of the n-th-order spatial differences and lower-order terms. An n-th-order array can also be written as the product of n first-order response terms as

$$y_n(\omega,\phi) = A_0 \prod_{i=1}^{n} \left(1 - \mathrm{e}^{-\mathrm{i}\omega(T_i + d_i/c\cos\phi)}\right) , \quad (50.46)$$

where the d_i relate to the microphone spacings, and the T_i relate to the chosen time delays. There is a design advantage in expressing the array response in terms of the products of first-order terms: it is now simple to represent higher-order systems as cascaded systems of lower order. Figure 50.11 shows how differential arrays can be constructed for up to second order. Extension of the design technique to higher orders is straightforward.

Values of T_i can easily be determined from the microphone spacing and the desired null angle. The ordering of T_i is not important as long as $\omega T_i \ll \pi$. If again it is assumed that $\boldsymbol{k}d_i \ll \pi$ and $\omega T_i \ll \pi$, then (50.46) can be approximated as

$$y_n(\omega,\phi) \approx A_0 \omega^n \prod_{i=1}^{n} (T_i + d_i/c\cos\phi) . \quad (50.47)$$

Equation (50.47) can be further simplified by making some simple substitutions for the arguments in the product term. Setting $\alpha_i = T_i/(T_i + d_i/c)$, then

$$y_n(\omega,\phi) \approx A_0 \omega^n \prod_{i=1}^{n} [\alpha_i + (1-\alpha_i)\cos\phi] . \quad (50.48)$$

If the product in (50.48) is expanded, a power series in $\cos\phi$ can be written for the response of the n-th-order array to an incident plane wave

$$\begin{aligned} &y_n(\omega,\phi) \\ &= A_0 G \omega^n \left(a_0 + a_1\cos\phi + a_2\cos^2\phi + \ldots \right. \\ &\quad \left. + a_n \cos^n\phi\right) , \end{aligned} \quad (50.49)$$

where the constant G is an overall gain factor. The only frequency-dependent term in (50.49) is ω^n. Thus the frequency response of an n-th-order differential array can be easily compensated by a low-pass filter whose frequency response is proportional to ω^{-n}. By choosing a structure that places only a delay behind each element in a differential array, the coefficients in the power series in (50.49) are independent of frequency, resulting in an array whose beampattern is independent of frequency.

To simplify the following exposition on the directional properties of differential arrays, it is assumed that the amplitude factor can be neglected. Also, since the directional pattern described by the power series in $\cos\phi$ can have any general scaling, the directional response can be written as solely a function of ϕ:

$$\begin{aligned} y_n(\phi) &= a_0 + a_1\cos\phi + a_2\cos^2\phi + \ldots \\ &\quad + a_n\cos^n\phi . \end{aligned} \quad (50.50)$$

Without loss of generality, the coefficients a_i can be defined to attain a normalized response at $\phi = 0°$, which implies

$$\sum_{i=0}^{n} a_i = 1. \quad (50.51)$$

In general, n-th-order differential microphones have, at most, n nulls (zeros). This follows directly from (50.50) and the fundamental theorem of algebra.

As an instructive example, let us examine the specific case for a second-order array,

$$y_2(\phi) = a_0 + a_1\cos\phi + a_2\cos^2\phi . \quad (50.52)$$

Equation (50.52) can also be factored into two first-order terms and written as

$$y_2(\phi) = [\alpha_1 + (1-\alpha_1)\cos\phi][\alpha_2 + (1-\alpha_2)\cos\phi] , \quad (50.53)$$

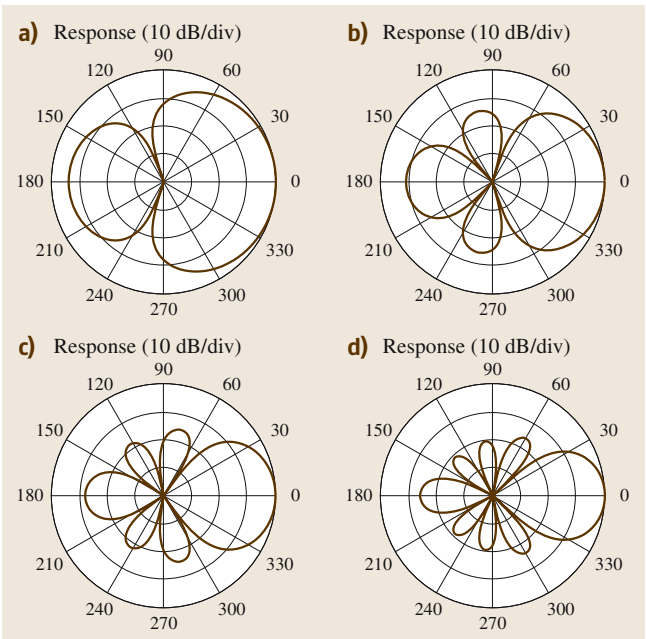


Fig. 50.12a–d Directional patterns that have maximum directional gain for differential microphone arrays for up to forth-order arrays

where

$$a_0 = \alpha_1\alpha_2,$$
$$a_1 = \alpha_1(1-\alpha_2)+\alpha_2(1-\alpha_1),$$
$$a_2 = (1-\alpha_1)(1-\alpha_2)\,, \tag{50.54}$$

or

$$\alpha_1 = a_0 + a_1/2 \pm \sqrt{(a_0+a_1/2)^2 - a_0}\,,$$
$$\alpha_2 = a_0 + a_1/2 \mp \sqrt{(a_0+a_1/2)^2 - a_0}\,. \tag{50.55}$$

As shown, the general form of the second-order system is the sum of second-, first-, and zero-order terms. If certain constraints are placed on the values of a_0 and a_1, it can be seen that there are two nulls (zeros) in the interval $0 \le \phi < \pi$. The array response pattern is axisymmetric about $\phi = 0$. These zeros can be explicitly found at the angles ϕ_1 and ϕ_2:

$$\phi_1 = \cos^{-1}\left(\frac{-\alpha_1}{1-\alpha_1}\right)\,, \tag{50.56}$$

$$\phi_2 = \cos^{-1}\left(\frac{-\alpha_2}{1-\alpha_2}\right)\,, \tag{50.57}$$

where now α_1 and α_2 can take on either positive or negative values where the magnitude of the inverse cosine is less than or equal to one. If the resulting beam pattern is constrained to have a maximum at $\phi = 0°$, then the values of α_1 and α_2 are also constrained. One interesting thing to note is that negative values of α_1 or α_2 correspond to a null moving into the front half-plane. Negative values of α_1 for the first-order microphone can be shown to have a rear-lobe sensitivity that exceeds the sensitivity at 0°. Since (50.53) is the product of two first-order terms, emphasis of the rear lobe caused by a negative value of α_2 can be counteracted by the zero from the term containing α_1. As a result, a beampattern can be found for the second-order microphone that has maximum sensitivity at $\phi = 0°$ and a null in the front-half plane. This result also implies that the beamwidth of a second-order microphone with a negative value of α_2 is narrower than that of the second-order dipole ($\cos^2(\phi)$ directional dependence). Figure 50.12 shows the beam patterns that have the highest attainable directivity for first through forth-order differential arrays [50.11].

50.5 Eigenbeamforming Arrays

The scalar acoustic pressure sound field obeys the Helmholtz equation [50.1], which states that the distribution of acoustic pressure and particle velocity on a surface uniquely defines the sound field within this surface if no sources or obstacles are enclosed within the surface. One can also use the Helmholtz equation to compute the external sound field if there are no obstacles or sources external to the measured surface. Thus, the interior and exterior sound field is uniquely determined if the sound pressure and particle velocity on a closed surface are known. Therefore it follows that only the sound pressure and particle velocity on a closed surface is required to capture all information of the sound field. This observation leads to a different beamformer implementation where, instead of forming beams in the classical sense by combining filtered individual microphone signals, one can use spatially decomposed *eigenbeams* to realize a general beamformer.

50.5.1 Spherical Array

Although the array surface can be any general surface, one can greatly simplify the analysis and array construction by using microphones placed on a rigid surface. Using a rigid body has the advantage that the radial particle velocity on its surface is zero. This greatly simplifies the general solution since only the sound pressure needs to be measured. Using a spherical geometry also simplifies the mathematics and allows for a design that puts equal weight on all directions due to its spherical symmetry. Theoretically many other shapes are possible, but the most amenable are shapes that are congruous to separable coordinate systems like cylindrical, oblate,

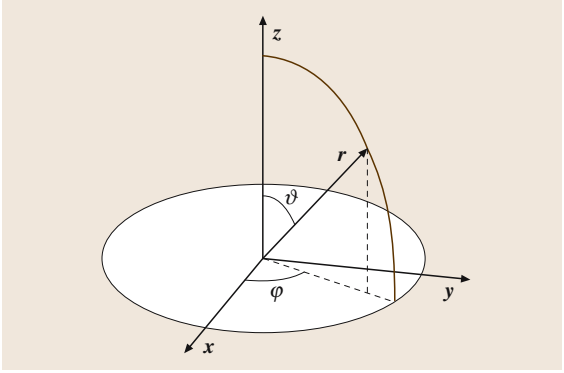


Fig. 50.13 Definition of the spherical coordinate system

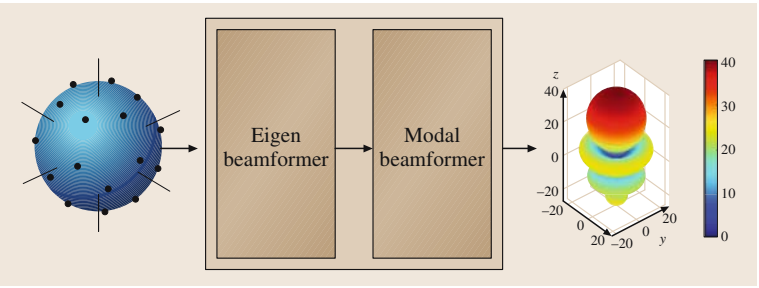


Fig. 50.14 Block diagram of the functional blocks of the spherical array

prolate, etc. Fig. 50.13 shows the notation for the standard spherical coordinate system that is used for the spherical array.

50.5.2 Eigenbeamformer

The overall structure of the spherical array can be decomposed into two main cascaded sections as shown in Fig. 50.14: an eigenbeamformer and a modal beamformer [50.15, 16].

The task of the eigenbeamformer is to transform the microphone signals into an orthonormal beam space. Since these beams are characteristic of the sound field in a similar way as eigenvectors are for a matrix, they are referred to as eigenbeams. Hence, the beamforming preprocessor is referred to as an eigenbeamformer. The second stage is a *modal* beamformer that linearly combines the eigenbeams to form a desired output beam pattern. In the following sections, it is shown that a spherical harmonic decomposition of the sound field leads to a computationally simple realization of the modal beamformer.

To begin an analysis of the spherical eigenbeamformer, we first need to introduce the orthonormal spherical harmonic eigenfunctions $Y_n^m(\vartheta, \varphi)$ which have order n and degree m, and are functions of the spherical angles ϑ, φ. The spherical harmonics Y_n^m are defined as [50.17],

$$Y_n^m(\vartheta, \varphi) = \sqrt{\frac{(2n+1)}{4\pi}\frac{(n-m)!}{(n+m)!}} P_n^m(\cos\vartheta)\mathrm{e}^{\mathrm{i}m\varphi}, \tag{50.58}$$

where $P_n^m(\cos\vartheta)$ are the standard associated Legendre functions. To calculate the response of the spherical eigenbeamformer to a planewave with incident angle ϑ, φ, and wavenumber k, it is judicious to express the plane wave in terms of spherical harmonics. By matching the boundary conditions of the rigid sphere of radius a to the impinging plane wave from the direction (ϑ, φ), one can compute the total acoustic pressure on the surface at location $(a, \vartheta_\mathrm{s}, \varphi_\mathrm{s})$ as

$$\begin{aligned} &p_\mathrm{s}(\vartheta_\mathrm{s}, \varphi_\mathrm{s}, ka, \vartheta, \varphi) \\ &= 4\pi \sum_{n=0}^{\infty} i^n b_n(ka) \sum_{m=-n}^{n} Y_n^m(\vartheta, \varphi) Y_n^{m^*}(\vartheta_\mathrm{s}, \varphi_\mathrm{s}) \,. \end{aligned} \tag{50.59}$$

The b_n are the *modal coefficients*, defined as

$$b_n(ka) = \left[j_n(ka) - \frac{j_n'(ka)}{h_n^{(2)'}(ka)} h_n^{(2)}(ka) \right], \tag{50.60}$$

where $j_n(ka)$ are the spherical Bessel functions, and the prime indicates the derivative with respect to its argument, and the function $h_n^{(2)}(k)$ is the spherical Hankel of the second kind.

From the previous development, it can be seen that the rigid spherical baffle has a unique pressure field for plane waves impinging from different angles. A desired beam in a specific direction can be formed by filtering the surface pressure by a general weighting of the acoustic pressure on the surface. The spherical eigenbeamformer operates by decomposing this general surface weighting function into spherical harmonics. Thus, the output of an eigenbeamforming microphone to an incident plane wave for a spherical harmonic weighting function of $Y_{n'}^{m'}(\vartheta_\mathrm{s}, \varphi_\mathrm{s})$ can be written

$$\begin{aligned} &F_{n',m'}(\vartheta, \varphi, ka) \\ &= \int_{\Omega_\mathrm{s}} p_\mathrm{s}(\vartheta_\mathrm{s}, \varphi_\mathrm{s}, ka, \vartheta, \varphi) Y_{n'}^{m'}(\vartheta_\mathrm{s}, \varphi_\mathrm{s}) \mathrm{d}\Omega_\mathrm{s}, \\ &= 4\pi i^{n'} b_{n'}(ka) Y_{n'}^{m'}(\vartheta, \varphi) \end{aligned} \tag{50.61}$$

where $\mathrm{d}\Omega_\mathrm{s}$ represents an integration over the surface of the sphere. The far-field directivity of the microphone has the same directional dependence as the applied spherical harmonic sensitivity function, namely $Y_{n'}^{m'}$. Thus, it can now be seen that forming a desired beam

can be accomplished by using spherical harmonics as the basis functions expansion of the general surface weighting and then appropriately weighting and summing the outputs of these basis functions. The paragraph below on eigenbeams gives a detailed analysis of these eigenbeams. One final thing to note here is that the modal coefficients b_n introduce a frequency dependance. In order to combine different eigenbeams appropriately to form a desired beam, the b_n coefficients must be equalized so that they have the same amplitude and frequency response.

Discrete Orthonormality

In order to realize a spherical array, the continuous aperture has to be sampled. To achieve the same result as obtained in (50.61), the sample positions must fulfill the following *discrete orthonormality* condition:

$$A_{nm} \sum_{s=0}^{S-1} Y_n^m(\vartheta_s, \varphi_s) Y_{n'}^{m'}(\vartheta_s, \varphi_s) = \delta_{nn'} \delta_{mm'} \; . \quad (50.62)$$

where the A_{nm} are scale factors that are required to fullfill the discrete orthonomality condition. Although is not trivial to find a set of sensor locations that fulfill this discrete orthonormality, one sensor arrangement that fulfills this constraint up to fourth-order spherical harmonics are sample points at the center of the 32 faces of a truncated icosahedron [50.16].

The resulting structure of the eigenbeamformer can be derived from (50.62): for a specific eigenbeam Y_n^m the microphone signals are weighted by the sampled values of the corresponding surface sensitivity, $Y_n^m(\vartheta_s, \varphi_s)$. It can therefore be seen that the eigenbeamformer is effectively a set of simple delay-sum beamformers per eigenbeam (with no delay required).

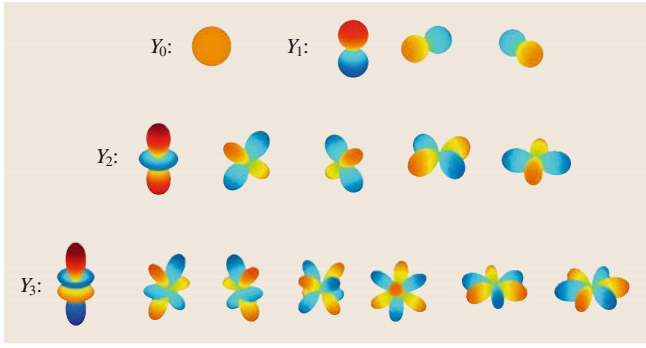


Fig. 50.15 Real and imaginary components of the spherical harmonics from zero to third order

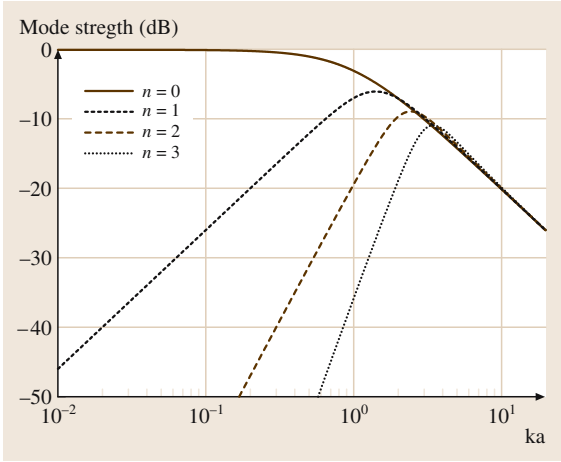


Fig. 50.16 Modal coefficients b_n for the first four orders

Besides the orthonormal constraint, spatial aliasing has to be taken into account when sampling a discrete aperture as in the design of classical beamforming arrays. Since there are $(N+1)^2$ spherical harmonics for a spatial resolution of order N, a minimum of $(N+1)^2$ sample locations are required to distinguish the spherical harmonics.

Eigenbeams

The outputs of the eigenbeamformer are a set of orthonormal beampatterns that are referred to as eigenbeams. These eigenbeams represent a spatially orthonormal decomposition of the sound field and represent a complete description of the original sound field up to the highest-order spherical harmonic used in the orthonormal expansion.

Figure 50.15 shows some example eigenbeams. From (50.58), it can be seen that the elevation dependence follows the Legendre function while the azimuth dependence has a sine-cosine dependance. The order n determines the number of zeros in the ϑ-direction while two times the degree m gives the number of zeros in the φ-direction.

The total number of eigenbeams required to capture N-th-order spherical harmonics of the sound field is $(N+1)^2$. As noted previously, the maximum achievable directional gain is $20\log(N+1)$. Thus, to obtain a maximum directional gain of 12 dB for any arbitrary direction, one would require eigenbeams up to third order. Note here that the term *order* has the same context as that used in differential arrays.

Similar to differential arrays, eigenbeams have the desirable quality that their directional response is fre-

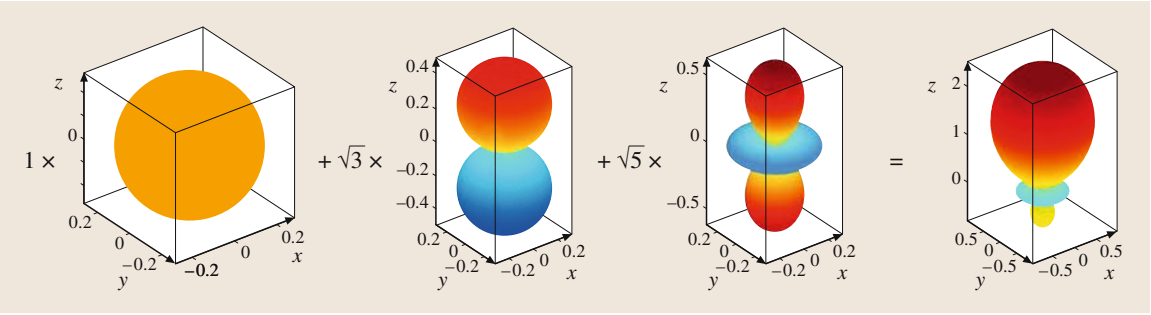


Fig. 50.17 Generating a second order hypercardioid pattern (note that only the directional properties are shown)

quency independent; however, as shown in Fig. 50.16 and (50.60), the magnitude response of the eigenbeams are dependent on the eigenbeam order and have high-pass responses that are $6n$ dB per octave. Thus one needs to equalize the different eigenbeams when combining them to realize a desired beampattern.

50.5.3 Modal Beamformer

The modal beamformer forms the second stage of the overall eigenbeamformer processing structure and comprises a simple weight-and-sum beamformer

$$D(\vartheta,\varphi)=\sum_{n=0}^{N} c_n Y_n(\vartheta,\varphi)\,, \tag{50.63}$$

where the beamformer multiplies each input beam by a factor c_n and adds up all weighted beams. As an example, Fig. 50.17 shows the generation of a second-order hypercardioid pattern steered along the z-axis.

To steer the resulting beam pattern towards a desired look-direction $[\vartheta_0,\varphi_0]$ one can use the Legendre polynomial addition theorem [50.1],

$$P_n(\cos\Theta) = \sum_{m=-n}^{n}\frac{(n-m)!}{(n+m)!}P_n^m(\cos\vartheta)P_n^m(\cos\vartheta_\mathrm{s})\,\mathrm{e}^{\mathrm{i}m(\varphi-\varphi_\mathrm{s})} \tag{50.64}$$

and (50.63) to write the overall steered array response as

$$D_{\vartheta_0\varphi_0}(\vartheta,\varphi) = \underbrace{\sum_{n=0}^{N} c_n}_{\text{combine}} \underbrace{\sum_{m=-n}^{n}\sqrt{\frac{(n-m)!}{(n+m)!}}P_n^m(\cos\vartheta_0)\,\mathrm{e}^{-\mathrm{i}m\varphi_0}}_{\text{steer}} \times Y_n^m(\vartheta,\varphi)\,. \tag{50.65}$$

Although modal beamforming arrays seem quite different from classical arrays, they are equivalent in operation but offer a more-intuitive and efficient way to realize beamforming arrays. As noted earlier, eigenbeamforming is not limited to using a spherical harmonic expansion to realize a modal beamformer, and any other spatially orthogonal basis function expansion can be used. Modal beamforming offers the capability of reducing the number of signals that need to be stored for processing since, in principal, the number of eigenbeams required can be much less than the number of microphone elements. Modal beamforming also has the desirable property that forming multiple simultaneous beams is computationally very efficient since the modal beamformer requires M multiply-adds, where M is the number of eigenbeams to form each unique output. Finally, the inherent computational simplicity of modal beamformers leads to efficient dynamic and adaptive microphone array implementations based on multiple beams and not on individual elements.

50.6 Adaptive Array Systems

The microphone arrays described in the previous sections are beamformers whose spatial responses are static since the weights, once designed, are fixed. Designs of fixed beamformers are done with assumptions of the signal and noise temporal and spatial statistics. This approach often leads to good designs when the a priori

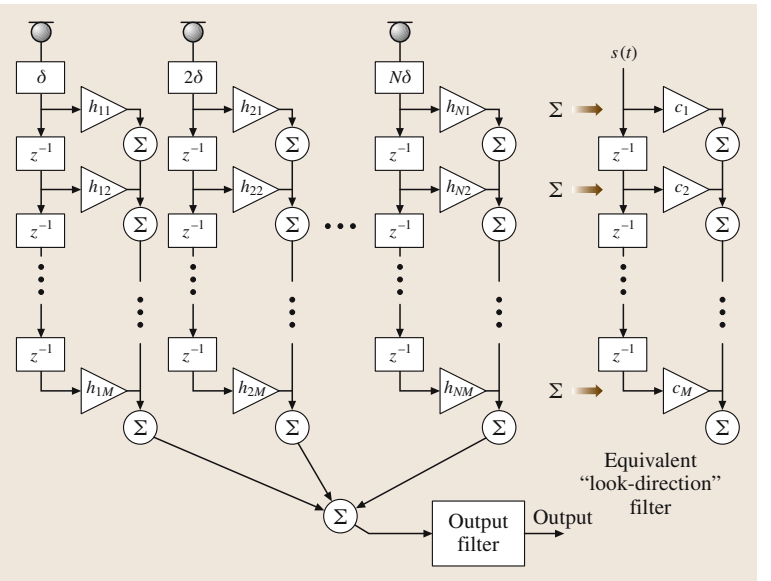


Fig. 50.18 Block diagram of a broadband adaptive microphone array implemented as an FIR filter-sum beamformer

assumptions match the acoustic conditions in which the array is placed. However, in acoustic and noise environments that dynamically change, it can be advantageous to use arrays that self-optimize their performance based on the sound field that the array is placed. Arrays that self-optimize their spatial response are generally referred to as adaptive arrays. There are many forms of adaptive array systems [50.18], but only a constrained broadband array is described here since most other schemes are based on similar approaches.

50.6.1 Constrained Broadband Arrays

Adaptive beamformers typically optimize a quadratic metric that is related to the output SNR. A common optimization is to minimize the output power under the constraint that the desired signal is not affected or only minor modifications are imparted to the signal. Thus one needs to minimize the output power under a constraint that the desired signal is not harmed. In most cases, like that of speech pickup in a room, the statistics of the desired source are not precisely known or are highly nonstationary. For this case, common adaptive array designs are based on the assumption that the position or direction of the desired source is known. The position can be found by beam-steering techniques based on power or known signal statistics, or by more sophisticated methods such as eigendecomposition [50.3] or other time-delay estimation direction-finding schemes [50.19]. It is further assumed that the locations of the desired source and noise sources change slowly. In a typical scenario, there may be a desired source at one angle and undesired interferers at other angles. In view of these requirements, adaptive optimization of the array output can result in better performance than designs based on assumed noise field models.

A block diagram of an adaptive filter-sum array is shown in Fig. 50.18. Each microphone feeds a weighted tapped delay line (FIR filter) whose intermediate outputs are summed. If there are N microphones and M taps on each delay line, there are a total of NM tap weights h_{nm}. All weights are adapted to optimize the output according to some specified criterion which, as mentioned above, is typically based on minimizing the output power under the constraint that the desired source direction is not affected or only slightly degraded.

The considerations governing the choice of N and M for a uniform linear array are described as follows. A linear combination of the microphone signals can produce at most $N-1$ controllable nulls in the directivity pattern of the array at a single frequency. If an M-tap FIR filter is placed behind each microphone, then the same number of nulls can be produced at M frequencies ($M/2$ if the weights are real). Thus, for broadband sources, N is governed by the expected number of noise sources and M by the frequency range over which the desired signal has appreciable energy. The goal of the adaptive algorithm is to search for the optimum location of the nulls under the constraints that signals arriving from the desired direction are not too distorted in frequency response.

Adaptive arrays of this type have been considered in the past. Constrained optimization of an adaptive array like the one shown in Fig. 50.18 can be written in matrix form as

$$\min_{\boldsymbol{h}} \boldsymbol{h}^{\mathrm{T}} \boldsymbol{R} \boldsymbol{h} \quad \text{subject to} \quad \boldsymbol{C}\boldsymbol{h} = \boldsymbol{c}\,, \tag{50.66}$$

where $\boldsymbol{R}$ is the input cross-correlation matrix, the $M \times N$ matrix $\boldsymbol{C}$ describes the M linearly independent constraints, and $\boldsymbol{c}$ is the set of M constraining values. *Frost* [50.20] developed an algorithm that minimizes the output power of the array while maintaining a flat magnitude and linear phase response for signals arriving from the desired source direction. The Frost algorithm uses a *hard constraint* such that the output contains only a delayed version of the signal arriving from the *look direction* (the look direction is the direction that is known a priori and is set by the delay values δ in Fig. 50.18). Constraints are enforced by noting that for signals propagating from the *look direction*, the signals moving through the FIR tapped-delay lines are all in phase. The matrix $\boldsymbol{C}$ sums all of the columns to form an equivalent vector $\boldsymbol{c}$ which is the response of the array to the look direction. The look-direction equivalent filter is the column sum and is shown in Fig. 50.18 as the rightmost FIR filter. For the Frost hard constraint of flat magnitude and linear phase, the vector $\boldsymbol{c}$ must be a Dirac delta function (where all taps but one tap are zero and the single nonzero tap is set to the desired gain of signals propagating from the look direction). Although the Frost algorithm is straightforward, one can extend the degrees of freedom by softening the constraint, which is useful when the number of taps is small.

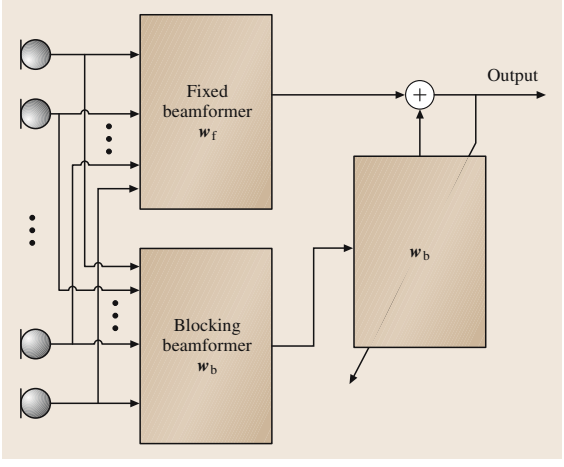


Fig. 50.19 Schematic of a generalized sidelobe canceler (GSC) adaptive array

One variant of the Frost beamformer that can simplify the computational cost by forcing the constraint into the front-end of the array processing, was proposed by *Griffiths* and *Jim* [50.21]. Their solution was to preform a set of $N-1$ difference beams that have nulls in the look direction. These *prebeams* are then used to adjust an adaptive filter-sum beamformer to minimize the total output power. Since the prebeams do not contain any look-direction signal, they implicitly invoke the look-direction constraint when combined adaptively with the main adaptive beamformer. Figure 50.19 shows a general schematic of how the Griffiths–Jim beamformer operates. In Fig. 50.19 the fixed filter-sum beamformer is denoted as $\boldsymbol{W}_{\mathrm{f}}$, the subtractive prebeamformer *blocking matrix* as $\boldsymbol{W}_{\mathrm{b}}$, and the adaptive beamformer as $\boldsymbol{W}_{\mathrm{a}}$. The general structure shown in Fig. 50.19 is known as a *generalized sidelobe canceler* (GSC) [50.18] and this structure was actually the basis of the first adaptive beamformers, which predated the development of the Frost beamformer. Intuitively, the GSC structure is appealing since it allows one to directly visualize the operation of the adaptive beamformer.

A GSC beamformer generally has one fixed main beam steered in the known look direction, and one or more secondary beams that have nulls in the look direction. These secondary beams are used to identify spatial noise sources not in the look-direction and then linearly combined with the main beam to adaptively minimize the output power or some other defined performance measure. Specific works on GSC-based beamformers for microphone arrays have been presented by *Hoshuyama* et al. [50.22], and *Herbordt* et al. [50.23].

It is well known that the Frost beamformer and GSC beamformers are sensitive to multiple paths [50.24], which manifest themselves as *signal cancelation*. Signal cancelation is due to correlated off-axis signals that cancel the desired look-direction signal by destructive interference within the filter-sum beamformer. This effect can be easily visualized by considering the operation of a GSC beamformer. In GSC beamforming, reflected look-direction signals pass the *blocking matrix* and can therefore be used by the adaptive beamformer to cancel the look-direction signal. The exact amount of signal cancelation depends on the geometry of the array as well as the acoustic environment. One practical implementation to mitigate this multipath effect is to use a smaller number of taps so that reflected signals do not enter into the adaptive FIR tapped-delay lines [50.24].

Kaneda and *Ohga* [50.25] replaced the Frost *hard* constraint with a softer constraint. Since they were

mainly interested in speech signals and realized that small perturbations to magnitude and phase were not detectable on a speech signal, their *soft* constraint allowed for some deviation in magnitude and phase for signals propagating from the look direction.

The *soft* constraint can be seen if one writes the transfer function $G(\mathrm{i}\omega)$ from the desired source to the output,

$$D = \int |G(\mathrm{i}\omega) - 1|^2 \, \mathrm{d}\omega \leq D_0, \tag{50.67}$$

where D_0 is some specified positive number.

Neither the Frost or Kaneda approaches exploited the fact that the quality of speech is less sensitive to the phase of the transfer function from the desired source direction to the array output. In order to capitalize on this property, a different soft-constraint algorithm was proposed by *Sondhi* et al. [50.24] that places a constraint on only the magnitude of the transfer function. This modified constraint results in an output magnitude response variation defined as

$$D = \int \left[|G(\mathrm{i}\omega)|^2 - 1 \right]^2 \mathrm{d}\omega. \tag{50.68}$$

Fortunately a computationally simple algorithm can be found that uses a gradient search to minimize the normalized output power with the constraint that

$$D \leq D_0 \, . \tag{50.69}$$

A constraint such as (50.69) is difficult to impose directly. Therefore a search to minimize another cost function C defined as

$$C = P + \epsilon D \tag{50.70}$$

leads to a solution, where P is the normalized output power and ϵ is a Lagrange multiplier. The value D obtained at the optimum setting of the filters is then implicitly a function of the parameter ϵ. By varying ϵ, a desired value of the distortion is realized.

Figure 50.18 contains a block labeled *output filter*. The purpose of this filter is to compensate for the frequency response deviation that occurs in the softly constrained adaptive beamformer. In order for the array to steer nulls at frequencies where wavelengths are longer than the array length, the array must resort to pattern differencing, which results in a commensurate high-pass response in the look direction. If one constrains the array too tightly, the adaptive algorithm focuses on maintaining flat array response and not on minimizing the output power. By allowing larger-magnitude distortion at lower frequencies, the adaptive algorithm focuses on minimizing the output power at these lower frequencies. The inverse filter acts to compensate the commensurate low-frequency roll-off distortion. An approach that accomplishes a similar result was introduced by *Gooch* and *Shynk* [50.26], who proposed an adaptive array containing both pole and zero filters. This can effectively handle the constrained adaptive array and the frequency response change due to superdirectional operation.

50.7 Conclusions

This chapter has introduced various types of beamforming microphone array systems and discussed some of the fundamental theory of their operation, design, and implementation. With the continuing trend of lower cost electronics, it is now feasible to implement microphone arrays in a variety of low-cost consumer audio communication devices. It was shown that appropriately designed microphone arrays have the ability to offer directional gains that can significantly improve the quality of hands-free speech pickup in reverberant and noisy environments.

References

50.1 P.M. Morse, K.U. Ingard: *Theoretical Acoustics* (McGraw Hill, New York 1968)

50.2 B.D. Steinberg: *Principles of Aperture and Array System Design* (Wiley, New York 1976)

50.3 D.H. Johnson, D.E. Dudgeon: *Array Signal Processing: Concepts and Techniques* (Prentice-Hall, Upper Saddle River 1993)

50.4 J.L. Flanagan, J.D. Johnston, R. Zahn, G.W. Elko: Computer-steered microphone arrays for sound transduction in large rooms, J. Acoust. Soc. Am. **78**, 1508–1518 (1985)

50.5 W. Kellermann: A self steering digital microphone array, Proc. IEEE ICASSP **1991**, 3581–3584 (1991)

50.6 R.J. Lustberg: *Acoustic Beamforming Using Acoustic Arrays* (MIT Masters Thesis, MIT Cambridge 1993)

50.7 T.C. Chou: *Broadband Frequency-Independent Beamforming* (MIT Masters Thesis, MIT Cambridge 1994)

50.8 M.M. Goodwin: *Implementation and Applications of Electroacoustic Array Beamformers* (MIT Masters Thesis, MIT Cambridge 1992)

50.9 D.B. Ward, R.A. Kennedy, R.C. Williamson: FIR filter design for frequency invariant beamformers, IEEE Signal Process. **3**, 69–71 (1996)

50.10 D.K. Cheng, F.I. Tseng: Gain optimization for arbitrarily antenna arrays, IEEE Trans. Antennas Prop. **AP-13**, 973–974 (1965)

50.11 G.W. Elko: Differential microphone arrays. In: *Audio Signal Processing for Next Generation Multimedia Communication Systems*, ed. by Y. Huang, J. Benesty (Kluwer Academic, Dordrecht 2004)

50.12 N. Yaru: A note on super-gain antenna arrays, Proc IRE **39**, 1081–1085 (1951)

50.13 E.N. Gilbert, S.P. Morgan: Optimum design of antenna arrays subject to random variations, Bell System Tech. J. **34**, 637–663 (1955)

50.14 H. Cox, R. Zeskind, T. Kooij: Practical supergain, IEEE Trans. ASSP **34**, 393–398 (1986)

50.15 J. Meyer, G.W. Elko: A highly scalable spherical microphone array based on an orthonormal decomposition of the soundfield, IEEE ICASSP **2002**, 1781–1784 (2002)

50.16 J. Meyer, G.W. Elko: Spherical Microphone Arrays for 3D Sound Recording. In: *Audio Signal Processing for Next Generation Multimedia Communication Systems*, ed. by Y. Huang, J. Benesty (Kluwer Academic, Dordrecht 2004)

50.17 E.G. Williams: *Fourier Acoustics* (Academic Press, San Diego 1999)

50.18 R.A. Monzingo, T.W. Miller: *Introduction to Adaptive Arrays* (Wiley, New York 1980)

50.19 L. Chen, J. Benesty, Y. Huang: Time delay estimation in room acoustic environments: an overview, EURASIP Appl. Signal Process. **2006**, ID26503 (1972)

50.20 O.L. Frost: An algorithm for linear constrained adaptive array processing, Proc. IEEE **60**, 926–935 (1972)

50.21 L.J. Griffiths, C.W. Jim: An alternative approach to linearly constrained beamforming, IEEE Trans. Antennas Prop. **AP-30**, 27–34 (1982)

50.22 O. Hoshuyama, A. Sugiyama, A. Hirano: A robust adaptive beamformer for microphone arrays with a blocking matrix using constrained adaptive filters, IEEE Trans. Signal Process. **47**, 2677–2684 (1999)

50.23 W. Herboldt, W. Kellermann: Analysis of blocking matrices for generalized sidelobe cancellers for non-stationary broadband signals, Proc. IEEE ICASSP **2002**, 4187–4191 (2002)

50.24 M.M. Sondhi, G.W. Elko: Adaptive optimization of microphone arrays under a nonlinear constraint, Proc. IEEE ICASSP **2002**, 981–984 (2002)

50.25 Y. Kaneda, J. Ohga: Adaptive microphone array system for noise reduction, IEEE Trans. ASSP **34**, 1391–1400 (1986)

50.26 R.P. Gooch, J.J. Shynk: Wide-band adaptive array processing using pole-zero digital filters, IEEE Trans. Antennas Prop. **AP-34**, 355–367 (1985)

53. Sound Field Reproduction

R. Rabenstein, S. Spors

Multichannel sound field reproduction aims at the physically correct synthesis of acoustical wave fields with a large number of loudspeakers. It goes beyond stereophony by extending or eliminating the so-called sweet spot. This chapter presents the physical and mathematical description of various methods for this purpose and presents the physical and mathematical background of several methods for multichannel sound field reproduction.

Section 53.2 introduces the mathematical representation of sound fields. The panning laws of stereophony are given in Sect. 53.3 as an introduction to vector-based amplitude panning in Sect. 53.4. Then mathematically more-involved methods are described in the Sections on ambisonics (Sect. 53.5) and wave field synthesis (Sect. 53.6).

53.1 Sound Field Synthesis

Sound field synthesis encompasses a number of techniques for the creation of acoustical events using electroacoustical transducers. The most widespread application is the reproduction of acoustical performances for human listeners. Another emerging application is the reproduction of spatially distributed environmental noise (engines, streets, car interior, etc.) in laboratory settings for testing of speech communication equipment. Since all techniques for sound field synthesis have been initially developed for human entertainment, only the first application is used as an example below.

The spatial sound impression perceived by human listeners depends on two major factors

- the sound pressure at the listener's ears
- the psychoacoustics of spatial hearing

The first factor is a physical quantity that can be produced with electroacoustical equipment. The second factor requires precise knowledge about how our hearing organs translate physical excitations into spatial perception. Various reproduction methods make use of these factors in different ways. They can be roughly classified

into binaural techniques, stereophony, and sound field synthesis.

Binaural techniques use head phones to create the required sound pressure immediately at the listener's ears. There is no need for a detailed analysis of acoustical waves. Instead the design of binaural techniques is based mostly on the psychoacoustics of spatial hearing.

Stereophony uses two or more loudspeakers to deliver the sound pressure to the listener. The sound field around the listener's head can be described as a superposition of the sound waves emitted by the loudspeakers subject to scattering at the head. Then psychoacoustics plays a similar role as for binaural techniques.

Sound field synthesis methods use a large number of loudspeakers (tens to hundreds) to reproduce a sound field not only at the ears of one listener but in a larger space enclosing multiple listeners. Here the focus is on the technology for the correct reproduction of a physical field quantity. Psychoacoustical effects play a minor role as long as it can be assumed the listeners respond to a synthesized sound field in the same way as to the original one.

53.2 Mathematical Representation of Sound Fields

Sound fields are represented by scalar- and vector-valued quantities that depend on two or three spatial and one time coordinate. This section discusses various coordinate systems and different representations of sound fields with respect to these coordinates.

53.2.1 Coordinate Systems

The evolution of sound fields with time is denoted by the scalar time coordinate t. The spatial distribution of sound fields requires in general a three-dimensional (3-D) coordinate system. However, many arrangements of microphones and loudspeakers are restricted to positions in a plane, e.g., a horizontal plane in the height of the listener's ears. Then two spatial dimensions are sufficient although sound propagation is still a three-dimensional process. In these cases, the term two-dimensional (2-D) sound field is used to denote a special case of a three-dimensional sound field that exhibits no dependence on one of the three spatial coordinates. Different representations of two- and three-dimensional sound fields are presented next.

In Cartesian coordinates the three-dimensional space coordinates are denoted by

$$\boldsymbol{z} = \begin{pmatrix} \boldsymbol{x} \\ z \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} , \tag{53.1}$$

where x and y are chosen for a two-dimensional representation of a three-dimensional function independent of z. These functions are also written $u(\boldsymbol{x}) = v(\boldsymbol{z})$.

For two spatial dimensions, also circular or polar coordinates are used here. Two-dimensional Cartesian coordinates $\boldsymbol{x}$ and polar coordinates $\boldsymbol{r}$

$$\boldsymbol{x} = \begin{pmatrix} x \\ y \end{pmatrix} , \qquad \boldsymbol{r} = \begin{pmatrix} r \\ \alpha \end{pmatrix} \tag{53.2}$$

are related by

$$\boldsymbol{x} = r \begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix} , \qquad \boldsymbol{r} = \begin{pmatrix} \sqrt{x^2 + y^2} \\ \arctan\left(\frac{y}{x}\right) \end{pmatrix} . \tag{53.3}$$

Figure 53.1 shows the coordinate systems discussed above. Functions of two-dimensional Cartesian and polar coordinates are distinguished by the subscripts 'c' and 'p', such that $u_\mathrm{c}(\boldsymbol{x}) = u_\mathrm{p}(\boldsymbol{r})$ holds.

53.2.2 Wave Equation

Physical descriptions of sound fields are usually formulated in terms of the scalar-valued sound pressure $p(\boldsymbol{z}, t)$ and the vector-valued particle velocity $\boldsymbol{v}(\boldsymbol{z}, t)$. Their mutual dependencies are described by certain partial differential equations: Newton's law of motion

$$\operatorname{grad} p(\boldsymbol{z}, t) + \rho \frac{\partial}{\partial t} \boldsymbol{v}(\boldsymbol{z}, t) = 0 \tag{53.4}$$

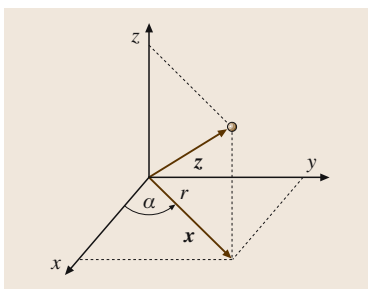


Fig. 53.1 Cartesian and polar coordinates in two and three dimensions

and the equation of continuity

$$\frac{\partial}{\partial t} p(\boldsymbol{z}, t) + \rho c^2 \operatorname{div} \boldsymbol{v}(\boldsymbol{z}, t) = 0 \,, \tag{53.5}$$

where ρ denotes the static density and c the speed of sound. Elimination of the particle velocity leads to the wave equation

$$\frac{\partial^2}{\partial t^2} p(\boldsymbol{z}, t) + c^2 \nabla^2 p(\boldsymbol{z}, t) = 0 \,, \tag{53.6}$$

which constitutes the basic mathematical representation of acoustic phenomena [53.1–4]. Solutions of the wave equation are called *sound fields* or *wave fields*.

The wave equation holds for different coordinate systems with the appropriate definition of the nabla operator ∇. Unique solutions also require physically meaningful boundary and initial conditions. Boundary conditions are determined by the acoustical properties of reflecting surfaces. Enclosures where reflections can be neglected are often modeled by free field conditions. Arbitrary solutions of the wave equation may be represented as a superposition of simple wave forms [53.4], e.g., plain or spherical waves as discussed below.

53.2.3 Plane Waves

A simple solution of the wave equation are the so-called plane waves. Their mathematical description in two and three spatial dimensions is now introduced.

Plane Waves in Three Dimensions

Consider the function

$$p(\boldsymbol{z}, t) = f(ct + \boldsymbol{n}^{\mathrm{T}} \boldsymbol{z}) = g\big(t + \frac{1}{c} \boldsymbol{n}^{\mathrm{T}} \boldsymbol{z}\big) \tag{53.7}$$

with $f(ct) = g(t)$ and a vector $\boldsymbol{n}$ of unit length. Vector transposition is indicated by the superscript 'T'. Equation (53.7) is a solution of the wave equation, as can be proven by inserting it into (53.6).

The arbitrary function $g(t)$ – and similarly $f(t)$ – describes the waveform of $p(\boldsymbol{z}, t)$ at the origin of the spatial coordinate system. The argument of f represents a plane in space with normal vector $\boldsymbol{n}$. As time increases, this plane propagates through space from the direction of $\boldsymbol{n}$ with speed c. Therefore the solution (53.7) is called a *plane-wave solution*.

Fourier transformation [53.5–7]

$$G(\omega) = \mathcal{F}\{g(t)\} = \int_{-\infty}^{\infty} g(t)\, \mathrm{e}^{-\mathrm{i}\omega t}\, \mathrm{d}t \tag{53.8}$$

of $p(\boldsymbol{z}, t)$ with respect to t gives

$$P(\boldsymbol{z}, \omega) = G(\omega)\, \mathrm{e}^{\mathrm{i}\boldsymbol{k}^{\mathrm{T}} \boldsymbol{z}} \tag{53.9}$$

with the wave vector $\boldsymbol{k} = k\boldsymbol{n}$ and the wave number $k = \omega/c$.

In this form, the influence of the temporal variation of the waveform $g(t)$ and the propagation through space are separated into two multiplicative factors. The spectral content of the waveform is given by $G(\omega)$ and the propagation of the plane wave is given by the exponential term.

Plane Waves in Two Dimensions

A plane wave in the xy-plane is characterized by the angle θ of the normal vector $\boldsymbol{n}$ from (53.7) in this plane

$$\boldsymbol{n} = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \,. \tag{53.10}$$

To indicate the direction from which the plane wave emanates, the angle θ is added to the list of arguments of the sound pressure $p_{\mathrm{c}}(\boldsymbol{x}, t, \theta)$ and the corresponding waveform $g(t, \theta)$ and likewise for their Fourier transforms $P_{\mathrm{c}}(\boldsymbol{x}, \omega, \theta)$ and $G(\omega, \theta)$.

To express two-dimensional plane waves in Cartesian coordinates

$$P_{\mathrm{c}}(\boldsymbol{x}, \omega, \theta) = G(\omega, \theta)\, \mathrm{e}^{\mathrm{i}\boldsymbol{k}^{\mathrm{T}} \boldsymbol{x}} \tag{53.11}$$

in polar form, it is convenient to separate the spectral properties $G(\omega, \theta)$ from the spatial properties as

$$P_{\mathrm{c}}(\boldsymbol{x}, \omega, \theta) = G(\omega, \theta)\, \hat{p}_{\mathrm{c}}(\boldsymbol{x}, \theta) \tag{53.12}$$

with

$$\hat{p}_{\mathrm{c}}(\boldsymbol{x}, \theta) = \hat{p}_{\mathrm{c}}(x, y, \theta) = \mathrm{e}^{\mathrm{i}\boldsymbol{k}^{\mathrm{T}} \boldsymbol{x}} \,. \tag{53.13}$$

A change of the spatial coordinate affects only the spatial term $\hat{p}_{\mathrm{c}}(x, y, \theta)$ but not $G(\omega, \theta)$ in (53.11). Therefore, only this term is considered for a transition to polar coordinates.

In polar coordinates, the wavevector $\boldsymbol{k}$ takes the form

$$\boldsymbol{k} = k\boldsymbol{n} = k \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \,. \tag{53.14}$$

The polar representation $\hat{p}_{\mathrm{c}}(x, y, \theta) = \hat{p}_{\mathrm{p}}(r, \alpha, \theta)$ follows by expressing $\boldsymbol{k}^{\mathrm{T}} \boldsymbol{x}$ through r and α

$$\boldsymbol{k}^{\mathrm{T}} \boldsymbol{x} = kr \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix} = kr \cos(\theta - \alpha) \tag{53.15}$$

as

$$\hat{p}_{\mathrm{p}}(r, \alpha, \theta) = \mathrm{e}^{\mathrm{i}kr \cos(\theta - \alpha)} \,. \tag{53.16}$$

Since $\hat{p}_\mathrm{p}(r, \alpha, \theta)$ is a periodic function of α, it can be expanded into a Fourier series. From the properties of the Bessel functions $J_\nu(kr)$ of the first kind and order ν [53.8–11] follows the Fourier series expansion (Jacobi–Anger expansion) [53.9]

$$\hat{p}_\mathrm{p}(r, \alpha, \theta) = \sum_{\nu=-\infty}^{\infty} \mathrm{i}^\nu J_\nu(kr)\, \mathrm{e}^{\mathrm{i}\nu(\alpha-\theta)} \,. \tag{53.17}$$

This can be regarded as an expansion of a plane wave into the angular modes $\mathrm{e}^{\mathrm{i}\nu\alpha}$ with coefficients $\mathrm{i}^\nu J_\nu(kr)\mathrm{e}^{-\mathrm{i}\nu\theta}$. This series can also be written in terms of sine and cosine functions. From the symmetry relation of the Bessel functions $J_\nu(kr)$ in the form

$$\mathrm{i}^{-\nu} J_{-\nu}(kr) = \mathrm{i}^\nu J_\nu(kr) \tag{53.18}$$

it follows that

$$\hat{p}_\mathrm{p}(r, \alpha, \theta) = J_0(kr) + 2\sum_{\nu=1}^{\infty} \mathrm{i}^\nu J_\nu(kr)\, \cos[\nu(\alpha-\theta)] \,. \tag{53.19}$$

A similar representation can be derived on the basis of sine functions.

With any of the representations (53.16), (53.17), or (53.19) and

$$\hat{p}_\mathrm{c}(\boldsymbol{x}, \theta) = \hat{p}_\mathrm{c}(x, y, \theta) = \hat{p}_\mathrm{p}(r, \alpha, \theta) = \hat{p}_\mathrm{p}(\boldsymbol{r}, \theta) \tag{53.20}$$

follows the two-dimensional plane wave in polar coordinates

$$P_\mathrm{p}(\boldsymbol{r}, \omega, \theta) = G(\omega, \theta)\, \hat{p}_\mathrm{p}(\boldsymbol{r}, \theta) \,, \tag{53.21}$$

$$P_\mathrm{p}(r, \alpha, \omega, \theta) = G(\omega, \theta)\, \hat{p}_\mathrm{p}(r, \alpha, \theta) \,. \tag{53.22}$$

The notation $P_\mathrm{p}(r, \alpha, \omega, \theta)$ is used to address the different components of $\boldsymbol{r}$ individually.

53.2.4 General Wave Fields in Two Dimensions

Representation by Angular Modes

Real wave fields rarely exhibit the ideal plane-wave property presented above. However, general wave fields can be represented as a superposition of plane waves from different directions θ and with different spectral weighting $G(\omega, \theta)$. This superposition follows from (53.22) by integration with respect to all possible directions $0 \le \theta < 2\pi$ as [53.4, 11]

$$P_\mathrm{p}(\boldsymbol{r}, \omega) = \frac{1}{2\pi} \int_0^{2\pi} G(\omega, \theta)\, \hat{p}_\mathrm{p}(r, \alpha, \theta)\, \mathrm{d}\theta \,, \tag{53.23}$$

where $P_\mathrm{p}(\boldsymbol{r}, \omega)$ denotes a general wave field with contributions from all directions θ.

Replacing $\hat{p}_\mathrm{p}(r, \alpha, \theta)$ by the expansion (53.17) gives

$$P_\mathrm{p}(\boldsymbol{r}, \omega) = \sum_{\nu=-\infty}^{\infty} \mathrm{i}^\nu J_\nu(kr)\, G^\circ(\omega, \nu)\, \mathrm{e}^{\mathrm{i}\nu\alpha} \,, \tag{53.24}$$

where $G^\circ(\omega, \nu)$ are the Fourier series coefficients

$$G^\circ(\omega, \nu) = \frac{1}{2\pi} \int_0^{2\pi} G(\omega, \theta)\, \mathrm{e}^{-\mathrm{i}\nu\theta}\, \mathrm{d}\theta \,. \tag{53.25}$$

Similar as for the plane wave in (53.17), (53.25) is an expansion into the angular modes with the coefficients $\mathrm{i}^\nu J_\nu(kr)\, G^\circ(\omega, \nu)$ and similar to (53.19) there is a representation in terms of the real and imaginary part of $G^\circ(\omega, \nu) = G^\circ_\mathrm{R}(\omega, \nu) + \mathrm{i}G^\circ_\mathrm{I}(\omega, \nu)$

$$\begin{aligned} P_\mathrm{p}(\boldsymbol{r}, \omega) = {} & J_0(kr)\, G^\circ(\omega, 0) \\ & + 2\sum_{\nu=1}^{\infty} \mathrm{i}^\nu J_\nu(kr)\, G^\circ_\mathrm{R}(\omega, \nu) \cos\nu\alpha \\ & - 2\sum_{\nu=1}^{\infty} \mathrm{i}^\nu J_\nu(kr)\, G^\circ_\mathrm{I}(\omega, \nu) \sin\nu\alpha \,, \end{aligned} \tag{53.26}$$

since the angular coefficients have conjugate symmetry $G^\circ(\omega, \pm\nu) = G^\circ_\mathrm{R}(\omega, \nu) \pm \mathrm{i}G^\circ_\mathrm{I}(\omega, \nu)$.

Physical Interpretation of the Angular Coefficients

The low-order angular coefficients have a physical interpretation in terms of sound pressure and particle velocity for $r = 0$. This is derived from (53.26) with the following relations for the Bessel functions

$$\begin{aligned} & J_0(0) = 1, \quad J_0'(0) = 0 \\ & J_1(0) = 0, \quad J_1'(0) = \tfrac{1}{2} \\ & J_\nu(0) = 0, \quad J_\nu'(0) = 0, \quad \nu > 1 \\ & \tfrac{\mathrm{d}}{\mathrm{d}r} J_\nu(kr) = k J_\nu'(kr) \qquad , \end{aligned} \tag{53.27}$$

where the prime denotes the derivative of a function with respect to its argument. Then the Fourier transform of the pressure $P_\mathrm{p}(\boldsymbol{r}, \omega)$ and its radial derivative $\partial/\partial r P_\mathrm{p}(\boldsymbol{r}, \omega)$ at $r = 0$ are given by

$$P_\mathrm{p}(\boldsymbol{r}, \omega)\big|_{r=0} = G^\circ(\omega, 0) \tag{53.28}$$

$$\begin{aligned} \frac{\partial}{\partial r} P_\mathrm{p}(\boldsymbol{r}, \omega)\Big|_{r=0} = {} & \mathrm{i}k\, G^\circ_\mathrm{R}(\omega, 1) \cos\alpha \\ & - \mathrm{i}k\, G^\circ_\mathrm{I}(\omega, 1) \sin\alpha \,. \end{aligned} \tag{53.29}$$

The real and imaginary part of the angular coefficient $G^\circ(\omega, 1)$ can also be expressed by the components of the

particle velocity in the x- and y-directions. The connection between sound pressure and particle velocity is established by transforming Newton's law of motion (53.4) for two spatial coordinates into the frequency domain

$$\operatorname{grad} P_{\mathrm{p}}(\boldsymbol{r},\omega) + \mathrm{i}kz_0 \boldsymbol{V}_{\mathrm{p}}(\boldsymbol{r},\omega) = 0 \tag{53.30}$$

with the free-field impedance $z_0 = \rho c$ and the Fourier transform of the particle velocity $\boldsymbol{V}_{\mathrm{p}}(\boldsymbol{r},\omega)$. Considering only the component in the radial direction gives

$$\frac{\partial}{\partial r} P_{\mathrm{p}}(\boldsymbol{r},\omega) + \mathrm{i}kz_0 V_{\mathrm{p},r}(\boldsymbol{r},\omega) = 0\,, \tag{53.31}$$

where $V_{\mathrm{p},r}(\boldsymbol{r},\omega)$ is the radial component of $\boldsymbol{V}_{\mathrm{p}}(\boldsymbol{r},\omega)$.

For $r = 0$ and $\alpha = 0,\ \frac{\pi}{2}$, it follows [using the notation from (53.22)]

$$\left.\frac{\partial}{\partial r} P_{\mathrm{p}}(r,0,\omega)\right|_{r=0} = -\mathrm{i}kz_0 V_{\mathrm{p},r}(0,0,\omega)\,,$$
$$\left.\frac{\partial}{\partial r} P_{\mathrm{p}}(r,\frac{\pi}{2},\omega)\right|_{r=0} = -\mathrm{i}kz_0 V_{\mathrm{p},r}(0,\frac{\pi}{2},\omega)\,.$$

Figure 53.1 shows that the radial components of the particle velocity $V_{\mathrm{p},r}(0,\alpha,\omega)$ in the direction $\alpha = 0$ and $\alpha = \frac{\pi}{2}$ are equal to the x- and y-components of $\boldsymbol{V}_{\mathrm{c}}(\boldsymbol{x},\omega)$ at the origin of the Cartesian coordinate system

$$\left.\frac{\partial}{\partial r} P_{\mathrm{p}}(r,0,\omega)\right|_{r=0} = -\mathrm{i}kz_0 V_{\mathrm{c},x}(0,0,\omega) \tag{53.32}$$
$$\left.\frac{\partial}{\partial r} P_{\mathrm{p}}(r,\frac{\pi}{2},\omega)\right|_{r=0} = -\mathrm{i}kz_0 V_{\mathrm{c},y}(0,0,\omega)\,. \tag{53.33}$$

Now the real and imaginary part of $G^{\circ}(\omega,1)$ can be identified from (53.29) with the x- and y-components of $\boldsymbol{V}_{\mathrm{c}}(0,0,\omega)$ as

$$z_0 V_{\mathrm{c},x}(0,0,\omega) = -G^{\circ}_{\mathrm{R}}(\omega,1) \tag{53.34}$$
$$z_0 V_{\mathrm{c},y}(0,0,\omega) = G^{\circ}_{\mathrm{I}}(\omega,1)\,. \tag{53.35}$$

Equations (53.28), (53.34), and (53.35) show the following physical interpretation of the angular coefficients for $\nu = 0, 1$: $G^{\circ}(\omega,0)$ is equal to the Fourier transform of the sound pressure and the real and imaginary parts of $G^{\circ}(\omega,1)$ are equal (up to a sign change) to the Fourier transforms of the x- or y-components of the particle velocity, where both sound pressure and particle velocity are measured at the origin of the coordinate system.

53.2.5 Spherical Waves

Arbitrary solutions of the wave equation with homogeneous boundary conditions are described in terms of Green's functions, which can be regarded as the response of a sound field to an impulse in time and space. They are the equivalent to impulse responses for one-dimensional (1-D) systems.

The Green's function $G_{3\mathrm{D}}(\boldsymbol{z}|\boldsymbol{z}',\omega)$ describes the contribution of a point source at position $\boldsymbol{z}'$ to the sound field at position $\boldsymbol{z}$. The position $\boldsymbol{z}$ is also referred to as the listener position. In the three-dimensional free field it is given by [53.12]

$$G^{\mathrm{f}}_{3\mathrm{D}}(\boldsymbol{z}|\boldsymbol{z}',\omega) = \frac{1}{4\pi}\,\frac{\mathrm{e}^{-\mathrm{i}\omega\|\boldsymbol{z}-\boldsymbol{z}'\|/c}}{\|\boldsymbol{z}-\boldsymbol{z}'\|}\,, \tag{53.36}$$

where $\|\boldsymbol{z}-\boldsymbol{z}'\|$ denotes the Euclidian distance between $\boldsymbol{z}$ and $\boldsymbol{z}'$. This Green's function describes a spherical wave and is also called the free-field Green's function. The denominator accounts for the decay of the amplitude over distance and the exponential term accounts for the time delay of the propagating spherical wave. Other forms of Green's functions exist for other types of spatial impulses, e.g., line sources.

53.2.6 The Kirchhoff–Helmholtz Integral

The *Kirchhoff–Helmholtz integral* or *Helmholtz integral equation* provides the relation between the sound field inside a volume of arbitrary shape and on the enclosing boundary. It is presented here because it is the key element of the wave field synthesis principle.

The Kirchhoff–Helmholtz integral expresses the sound pressure $P(\boldsymbol{z},\omega)$ inside a volume V by an integral

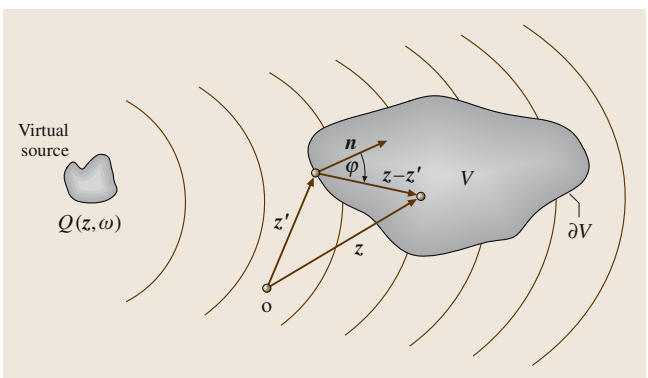


Fig. 53.2 Reproduction of the spatial wave field emitted by the virtual source $Q(\boldsymbol{z},\omega)$. The coordinate vector $\boldsymbol{z}$ addresses any arbitrary point inside the volume V, while $\boldsymbol{z}'$ denotes points on the boundary ∂V. $\boldsymbol{n}$ is the normal vector on ∂V and φ the angle between $\boldsymbol{n}$ and $\boldsymbol{z}-\boldsymbol{z}'$

on the surface ∂V [53.2,4,12,13]

$$\oint_{\partial V} P(z',\omega)\,\frac{\partial}{\partial \boldsymbol{n}} G_{3\mathrm{D}}(z|z',\omega)\,\mathrm{d}z' - \oint_{\partial V} G_{3\mathrm{D}}(z|z',\omega)\,\frac{\partial}{\partial \boldsymbol{n}} P(z',\omega)\,\mathrm{d}z' = \begin{cases} P(z,\omega)\,, & z \in V \\ 0\,, & z \notin V \end{cases} . \tag{53.37}$$

$G_{3\mathrm{D}}(z|z',\omega)$ is a Green's function that satisfies suitable homogeneous boundary conditions on ∂V. Figure 53.2 explains the spatial coordinates.

The Kirchhoff–Helmholtz integral states that at any point within the source-free region V the sound pressure $P(z,\omega)$ can be calculated if both the sound pressure $P(z',\omega)$ and its directional gradient $\frac{\partial}{\partial \boldsymbol{n}} P(z',\omega)$ are known on the boundary ∂V enclosing the volume. The direction $\boldsymbol{n}$ of the gradient points inward from the boundary ∂V. This boundary does not necessarily have to be a real physical existing surface.

53.3 Stereophony

Stereophony comprises a number of techniques for reproduction with two or more loudspeakers. They describe how to feed the loudspeakers with appropriate signals to create suitable spatial cues at the ears of a listener. These cues are expressed as interaural level and time differences and by other psychoacoustic principles [53.14]. Overviews on stereophonic techniques are given, e.g., in [53.15–17].

Stereophonic sound reproduction is covered only briefly here. The positioning of a phantom source between two loudspeakers is addressed on the basis of a purely physical description of the created sound field. Such an approach is not capable of describing all psychoacoustic effects encountered in practice. It merely serves as an introduction to the multichannel methods in the subsequent Sections.

Figure 53.4 shows a listener in the center of a coordinate system and two loudspeakers in the direction of the angles θ_0 and $-\theta_0$ to the right and left. The distance of the loudspeakers is not given explicitly, since it is assumed to be large enough that their sound fields at the position of the listener can be well approximated by a plane wave.

To reproduce the image of a sound source at some angle θ with $|\theta| < \theta_0$, a technique called *amplitude panning* is applied. The same driving signal is fed to both loudspeakers, but with different weighting factors g_r and g_l. These are selected such that the superposition of the sound fields of both loudspeakers makes the listener perceive a single sound source at the desired angle θ. This perception is called a *phantom source* or *virtual source*. The functional dependency of the weighting factors $g_\mathrm{r}(\theta)$ and $g_\mathrm{l}(\theta)$ on θ is called a *panning law*.

The choice of a suitable panning law depends not only on the acoustics of the sound field but also on human spatial perception. Therefore, a unique determination of the panning law is not possible and different proposals exist in the literature [53.17]. Two of these panning laws are presented here briefly: the sine law and the tangent law.

53.3.1 Sine Law

The sine panning law is based on the model of a locally plane wave. The weighting factors are determined such that the wave field in the close vicinity of the listener can be approximated by a plane wave with a normal vector in the desired direction.

To this end consider first only the right loudspeaker. Feeding it with an arbitrary time signal produces a spherical wave. If the loudspeaker is sufficiently far away from the listener, then the spherical wave can be approximated by a plane wave at the origin of the form of (53.9). Further it is assumed that the origin of the time axis is adjusted such that any delay terms due to the propagation delay from the loudspeaker to the listener vanish. The component of $P(z,\omega)$ in the direction of the unit vector $\boldsymbol{e}_x = [1\,0\,0]^\mathrm{T}$ in the x-direction is given by

$$P_{\mathbf{c},x}(x,\omega) = G(\omega,\theta_0)\,\mathrm{e}^{\mathrm{i}\boldsymbol{k}^\mathrm{T}x\boldsymbol{e}_x} = G(\omega,\theta_0)\,\mathrm{e}^{\mathrm{i}k_x x}\,. \tag{53.38}$$

where $k_x = k\sin\theta_0$ is the component of the wave vector $\boldsymbol{k}$ in the x-direction. Equation (53.38) represents a plane wave since the argument of the exponential function is linear in x.

Now consider the superposition of the sound fields of the two speakers fed by the same signal $G(\omega,\theta_0) = G(\omega,-\theta_0)$, but with an individual weighting factor g_r and g_l for the right and left loudspeaker, respectively.

The resulting sound field is

$$P_{\mathbf{c},x}(x,\omega) = G(\omega,\theta_0)\hat{p}_{\mathbf{c},x}(x) \tag{53.39}$$

with

$$\begin{aligned}\hat{p}_{\mathbf{c},x}(x) &= g_\mathrm{r}\,\mathrm{e}^{\mathrm{i}k_x x} + g_\mathrm{l}\,\mathrm{e}^{-\mathrm{i}k_x x} \\ &= (g_\mathrm{r}+g_\mathrm{l})\cos k_x x + j(g_\mathrm{r}-g_\mathrm{l})\sin k_x x \\ &= |\hat{p}_{\mathbf{c},x}(x)|\cdot \mathrm{e}^{\mathrm{i}\varphi(x)}\end{aligned} \tag{53.40}$$

The phase term $\varphi(x)$ of $\hat{p}(x)$ is not linear in x

$$\varphi(x) = \arctan\left[\frac{g_\mathrm{r}-g_\mathrm{l}}{g_\mathrm{r}+g_\mathrm{l}}\tan(k_x x)\right]. \tag{53.41}$$

However, close to the origin the arguments of both the tangent and the arc tangent are small and both functions may be approximated by their arguments. Then $\varphi(x)$ becomes, with $k_x = k\sin\theta_0$,

$$\varphi(x,\omega) \approx kx\left(\frac{g_\mathrm{r}-g_\mathrm{l}}{g_\mathrm{r}+g_\mathrm{l}}\sin\theta_0\right). \tag{53.42}$$

This expression is linear in x and thus represents a plane wave in the region where the approximation holds. According to Fig. 53.3 this is the area around the listeners head. Denoting the normal direction of this plane wave with θ, its corresponding angle is given by

$$\sin\theta = \frac{g_\mathrm{r}-g_\mathrm{l}}{g_\mathrm{r}+g_\mathrm{l}}\sin\theta_0 \tag{53.43}$$

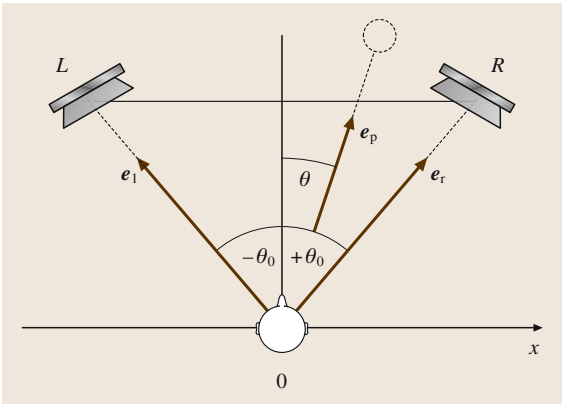


Fig. 53.3 Listener in front of two loudspeakers for two-channel stereophonic reproduction. The angle between the loudspeakers and the look direction of the listener is denoted by $\pm\theta_0$. The angle of a phantom source (*dashed lines*) relative to the listener is given by θ. The unit vectors in the direction of the loudspeakers and the phantom source are $\boldsymbol{e}_\mathrm{l}$, $\boldsymbol{e}_\mathrm{r}$, and $\boldsymbol{e}_\mathrm{p}$, respectively

for gain values $0 \le g_\mathrm{r}, g_\mathrm{l} \le 1$. The resulting panning law

$$\frac{g_\mathrm{r}-g_\mathrm{l}}{g_\mathrm{r}+g_\mathrm{l}} = \frac{1-\frac{g_\mathrm{l}}{g_\mathrm{r}}}{1+\frac{g_\mathrm{l}}{g_\mathrm{r}}} = \frac{\sin\theta}{\sin\theta_0} \tag{53.44}$$

is called the *sine law of stereophony* [53.16], and allows one to calculate the relationship between the gain factors g_r, g_l.

53.3.2 Tangent Law

The tangent panning law is based on a projection of the phantom source at angle θ to a coordinate system represented by the unit vectors $\boldsymbol{e}_\mathrm{r}$ and $\boldsymbol{e}_\mathrm{l}$ in the direction of the right and left loudspeaker (Fig. 53.3). These unit vectors do not need to be orthogonal but they must not be collinear. The gain factors for the left and right loudspeaker are then found as the corresponding projection coefficients.

The panning law follows from representing the unit vector in the direction of the virtual source $\boldsymbol{e}_\mathrm{p}$ by $\boldsymbol{e}_\mathrm{r}$ and $\boldsymbol{e}_\mathrm{l}$

$$\begin{aligned}\begin{pmatrix}\sin\theta\\ \cos\theta\end{pmatrix} &= g_\mathrm{l}\boldsymbol{e}_\mathrm{l} + g_\mathrm{r}\boldsymbol{e}_\mathrm{r} \\ &= g_\mathrm{l}\begin{pmatrix}\sin(-\theta_0)\\ \cos(-\theta_0)\end{pmatrix} + g_\mathrm{r}\begin{pmatrix}\sin\theta_0\\ \cos\theta_0\end{pmatrix} \\ &= \begin{pmatrix}(g_\mathrm{r}-g_\mathrm{l})\sin\theta_0\\ (g_\mathrm{r}+g_\mathrm{l})\cos\theta_0\end{pmatrix}\end{aligned} \tag{53.45}$$

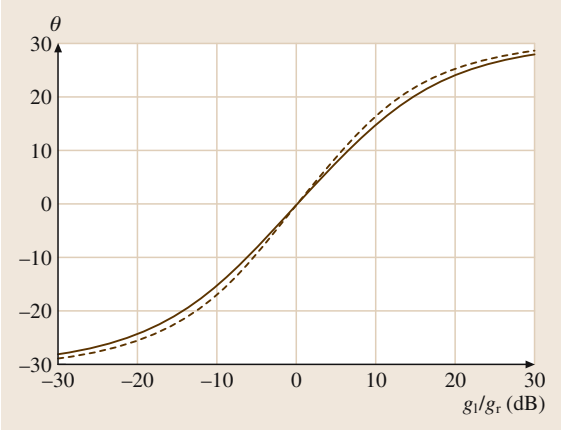


Fig. 53.4 Location θ of a phantom source versus the relation of the gain values $g_\mathrm{l}/g_\mathrm{r}$ for the sine (*solid line*) and tangent (*dashed line*) panning laws and for $\theta_0 = 30°$ (Fig. 53.3)

and by division of $\sin\theta$ through $\cos\theta$

$$\frac{g_\mathrm{r}-g_\mathrm{l}}{g_\mathrm{r}+g_\mathrm{r}}=\frac{1-\frac{g_\mathrm{l}}{g_\mathrm{r}}}{1+\frac{g_\mathrm{l}}{g_\mathrm{r}}}=\frac{\tan\theta}{\tan\theta_0}. \tag{53.46}$$

Figure 53.4 shows the angle θ of a phantom source depending on the relation of the gain coefficients $g_\mathrm{l}/g_\mathrm{r}$ according to the sine and tangent panning laws and a loudspeaker angle $\theta_0 = 30°$. Obviously, there are only minor differences between the panning laws.

53.3.3 Application of Amplitude Panning

The derivation of these two panning laws is based on idealized physical arguments and does not take into account the human spatial perception. However, it has been confirmed by experiments that not only does amplitude panning lead to the perception of a phantom source between the loudspeakers, but also that the relation between the perceived source angle and the gain factors is described reasonably well by the panning laws (see [53.16, Fig. 3.4]). For a discussion of the application of these panning laws in two- or three-channel stereo see [53.18].

There are two basic methods for the determination of the weighting factors g_r and g_l, either by the sound engineer at the mixing console or through an appropriate microphone setup for the recording of an auditory event.

Mixing consoles usually contain a device called panorama potentiometer (panpot). It is fed with a mono signal $g(t)$ and creates two weighted outputs $g_\mathrm{r}g(t)$ and $g_\mathrm{l}g(t)$. The values of g_r and g_l are not determined numerically. Rather the sound engineer varies the position of the panorama potentiometer until the position of the virtual source meets his intentions.

In live recording of auditory events the gain factors are created by the directional sensitivity of microphones. Consider placing two directional microphones almost at the same spot in such a way that the direction of their maximum sensitivity points in different directions. Then both microphones will record the same sound signal, however with different amplitude, i. e., with different gain factors. A certain spacing of the microphones introduces a delay between the microphone signals which can augment the spatial impression at the side of the listener. In practice, the choice of the directional microphones and their placement and orientation depends not on analytical formulas but on the art of the Tonmeister [53.17].

The panning laws presented here for two-channel stereo apply also to other stereophonic reproduction formats, such as surround sound or the so-called 5.1 format. However, not all additional loudspeakers are used for enhancing the spatial localization of virtual sound sources. The 5.1 format uses one channel for low-frequency effects with no spatial localization capabilities and the two rear loudspeakers for ambiance effects. The localization properties of the remaining three front loudspeakers are similar to two-channel stereo, except that the center speaker enhances the stability of the front image [53.16].

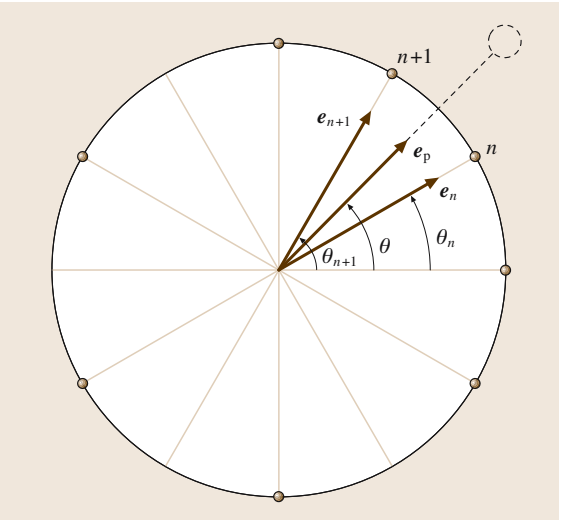


Fig. 53.5 Loudspeaker arrangement for two-dimensional vector-based amplitude panning. The angle between the loudspeaker n and an arbitrary reference direction is denoted by θ_n. The angle of a phantom source (*dashed lines*) is given by θ

53.3.4 The Sweet Spot

The derivation of the panning laws relies on some idealizations. It has been assumed that there is only one listener and that the position of the listener does not change. This ideal position which corresponds the origin of the coordinate system in Fig. 53.3 is called the *sweet spot*. Outside of this distinguished position, the assumptions for the panning laws do not hold. In addition, reflections in the listening rooms have been neglected.

For listeners outside of the sweet spot, the localization of virtual sources degrades. Close to the left or right loudspeaker, all sources are perceived at the location of this loudspeaker. The only way to avoid this problem is to increase the number of independent reproduction channels significantly. The multichannel audio reproduction methods to be discussed next, aim at either enlarging the sweet spot or extending the correct localization to the whole listening area.

53.4 Vector-Based Amplitude Panning

Vector-based amplitude panning (VBAP) is a multichannel audio reproduction method that uses amplitude panning just like in stereophony. The difference is that panning is not only applied to two loudspeakers, but to two or three adjacent loudspeakers out of an arbitrary number distributed on a circle or a sphere around the listener. In this way, virtual sources can appear on a full horizontal circle or anywhere in space. The theory and application of vector-based amplitude panning was first described in [53.19–23].

Vector-based amplitude panning extends the tangent panning law for two loudspeakers to panning between adjacent speakers of a one- or two-dimensional loudspeaker array. To express this extension in mathematical terms, reconsider the tangent panning law from Fig. 53.3. The projection of the unit vector $\boldsymbol{e}_\mathrm{p}$ in the direction of the virtual source can be written in vector form as

$$\boldsymbol{e}_\mathrm{p} = \boldsymbol{M}\boldsymbol{g} \tag{53.47}$$

with

$$\boldsymbol{e}_\mathrm{p} = \begin{pmatrix} \sin\theta \\ \cos\theta \end{pmatrix}, \qquad \boldsymbol{g} = \begin{pmatrix} g_\mathrm{l} \\ g_\mathrm{r} \end{pmatrix}, \tag{53.48}$$

$$\boldsymbol{M} = (\boldsymbol{e}_\mathrm{l}\ \ \boldsymbol{e}_\mathrm{r}) = \begin{pmatrix} -\sin\theta_0 & \sin\theta_0 \\ \cos\theta_0 & \cos\theta_0 \end{pmatrix}. \tag{53.49}$$

The gain factors follow from

$$\boldsymbol{g} = \boldsymbol{M}^{-1}\boldsymbol{e}_\mathrm{p} \tag{53.50}$$

after some trigonometric manipulations as

$$g_\mathrm{l} = \frac{\sin(\theta-\theta_0)}{\sin(2\theta_0)}, \quad g_\mathrm{r} = \frac{\sin(\theta+\theta_0)}{\sin(2\theta_0)}. \tag{53.51}$$

Thus the gain factors are the relation of the sine values of two angles: the angle between the virtual source and the respective loudspeaker ($\theta \pm \theta_0$) and the angle between both loudspeakers ($2\theta_0$).

53.4.1 Two-Dimensional Vector-Based Amplitude Panning

The determination of the gain factors for two-dimensional vector-based amplitude panning can be obtained in the same way. Figure 53.5 shows a general loudspeaker setup in a plane. Each loudspeaker is characterized by its angle θ_n relative to the center. The distance of the loudspeakers from the center does not matter since it is assumed that it is large enough that the wavefront from each loudspeaker at the center can be approximated by a plane wave. Consequently the virtual source is also only determined by its angle θ.

Now assume that the loudspeakers n and $n+1$ shall reproduce weighted signals to create a virtual source in between, i. e., $\theta_n \le \theta \le \theta_{n+1}$. The weighting factors g_n and g_{n+1} are obtained from a set of equations similar to (53.47)

$$\boldsymbol{e}_\mathrm{p} = \boldsymbol{M}_n\boldsymbol{g}_n \tag{53.52}$$

with

$$\boldsymbol{M}_n = \begin{pmatrix} \sin\theta_n & \sin\theta_{n+1} \\ \cos\theta_n & \cos\theta_{n+1} \end{pmatrix}, \ \boldsymbol{g}_n = \begin{pmatrix} g_n \\ g_{n+1} \end{pmatrix} \tag{53.53}$$

as

$$g_n = \frac{\sin(\theta_{n+1}-\theta)}{\sin(\theta_{n+1}-\theta_n)}, \quad g_{n+1} = \frac{\sin(\theta-\theta_n)}{\sin(\theta_{n+1}-\theta_n)}. \tag{53.54}$$

Now consider the case that the position of the virtual source moves without being restricted to certain loudspeaker positions, i. e., $0 \le \theta < 2\pi$. Then only those two loudspeakers are active which enclose the direction θ of the virtual source. For $\theta = \theta_n$ only one loudspeaker is active ($g_n = 1$).

In detail, the weighting factors g_ν for two-dimensional vector-based amplitude panning with N loudspeakers are given by

$$g_\nu(\theta) = \begin{cases} \frac{\sin(\theta_{n+1}-\theta)}{\sin(\theta_{n+1}-\theta_n)} & \nu = n \\ \frac{\sin(\theta-\theta_n)}{\sin(\theta_{n+1}-\theta_n)} & \nu = n+1 \\ 0 & \text{otherwise} \end{cases}. \tag{53.55}$$

Here, n denotes the current position of the virtual source such that $\theta_n \le \theta \le \theta_{n+1}$.

Just as the angle θ is computed modulo 2π, so the indices n of the adjacent loudspeakers are computed modulo N. For more than one virtual source the loudspeaker signals are a superposition of the panned signals for each single source.

Figure 53.6 shows the weighting functions for a two-dimensional vector-based amplitude panning system with five loudspeakers. Obviously, two-dimensional vector-based amplitude panning is the same as stereo panning, only that the position of the pair of active loudspeakers moves with the sound source.

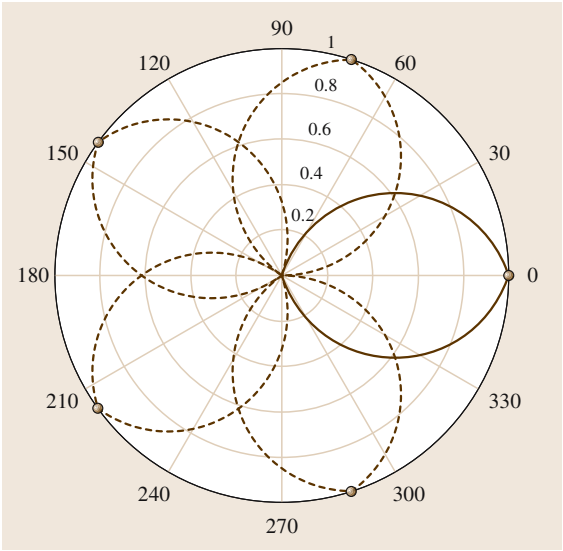


Fig. 53.6 Weighting coefficients for a two-dimensional vector-based amplitude panning system with five equally spaced loudspeakers

53.4.2 Three-Dimensional Vector-Based Amplitude Panning

The extension of vector-based amplitude panning to three spatial dimensions is straightforward. At first, a suitable loudspeaker configuration has to be chosen. As in the two-dimensional case, there is no special spatial distribution and no fixed number of loudspeakers. In [53.19], it is proposed to triangulate the sphere around the listener and to put one loudspeaker at each vertex. For any direction of a desired virtual source, there will be three loudspeakers forming a triangle which contains the source directions. As special cases the virtual source may lie between two loudspeakers or directly in the direction of one loudspeaker.

The virtual source is created by amplitude panning between the three loudspeakers of the corresponding triangle. The three weighting factors are again determined from a projection of the unit vector $\boldsymbol{e}_\mathrm{p}$ in the direction of the virtual source to the unit vectors $\boldsymbol{e}_1$, $\boldsymbol{e}_2$, $\boldsymbol{e}_3$ in the direction of the three loudspeakers. Solving the linear system of equations

$$\boldsymbol{e}_\mathrm{p} = \boldsymbol{M}\boldsymbol{g} \tag{53.56}$$

with

$$\boldsymbol{M} = \left(\boldsymbol{e}_1 \;\; \boldsymbol{e}_2 \;\; \boldsymbol{e}_3\right) \tag{53.57}$$

for the vector $\boldsymbol{g}$ yields the unknown weighting factors. The unit vectors $\boldsymbol{e}_\mathrm{p}$, $\boldsymbol{e}_1$, $\boldsymbol{e}_2$, and $\boldsymbol{e}_3$ have to be specified in a suitable three-dimensional coordinate system.

53.4.3 Perception of Vector-Based Amplitude Panning

So far, the virtual source has been considered as a well-defined subject. This assumption was useful to derive the various panning laws on a mathematical basis. Whether human listeners actually perceive a virtual source at the intended location can only be inferred from listening tests. Results reported in [53.22, 23] indicate that the panning laws for two- and three-dimensional vector-based amplitude panning do indeed correlate well with the human perception of the virtual source.

However, the localization of a virtual source depends on its targeted position relative to the adjacent speakers. The localization is most precise if the virtual source direction coincides with the position of a loudspeaker. In this case the loudspeaker is perceived as a real source. For virtual source position in between adjacent loudspeakers, a certain spread of the localization occurs. Methods to achieve a uniform spreading of amplitude panned virtual sources have been presented in [53.20].

53.5 Ambisonics

Ambisonics describes a family of methods that have been developed since the 1970s [53.24]. Like vector-based amplitude panning, Ambisonics uses a flexible number of loudspeakers in some distance of the listener such that the superposition of their wave fields can be regarded as a superposition of plane waves. In contrast to vector-based amplitude panning a virtual source is not only panned between two or three loudspeakers but by all loudspeakers in the arrangement. Ambisonics is also suitable for the reproduction of wave fields with a more-general structure than a superposition of a finite number of sources. In combination with suitable recording methods, Ambisonics allows the reproduction of real wave fields within the limits of certain approximations. Detailed descriptions of Ambisonics can be found in [53.25, 26].

53.5.1 Two-Dimensional Ambisonics

This section discusses the spatial reproduction with Ambisonics for planar loudspeaker arrays. First, panning of a plane wave from a certain direction is discussed, just as for stereophony and for vector based amplitude panning before. Then the reproduction of general wave fields is presented.

Panning Functions for the Reproduction of Plane Waves

Consider a two-dimensional loudspeaker setup as in Fig. 53.6. It is assumed that the loudspeakers produce sound waves from the directions θ_n that can be approximated by plane waves in the vicinity of the listeners head. A suitable superposition of these plane waves can approximate a virtual plane wave from an arbitrary direction θ. In contrast to vector-based amplitude panning not only two but all N loudspeakers contribute simultaneously to the superposition with appropriate weighting factors $g_n(\theta)$.

With the designation of the plane waves from different angles as in (53.11), this superposition is written as

$$P_\mathrm{p}(\boldsymbol{r}, \omega, \theta) = \sum_{n=0}^{N-1} P_\mathrm{p}(\boldsymbol{r}, \omega, \theta_n) \,. \tag{53.58}$$

The driving signal of each loudspeaker at θ_n is the waveform of the desired plane wave $g(t, \theta)$ multiplied by the weighting factor $g_n(\theta)$. The corresponding Fourier transforms are then related by

$$G(\omega, \theta_n) = g_n(\theta)\, G(\omega, \theta) \,. \tag{53.59}$$

Inserting (53.22) for θ and θ_n and (53.59) into (53.58) shows that $G(\omega, \theta)$ cancels and thus the panning functions $g_n(\theta)$ can be determined independently of the waveform $g(t, \theta)$

$$\hat{p}_\mathrm{p}(r, \alpha, \theta) = \sum_{n=0}^{N-1} g_n(\theta)\, \hat{p}_\mathrm{p}(r, \alpha, \theta_n) \,. \tag{53.60}$$

To derive conditions for the determination of the weighting factors, the functions in (53.60) are expanded into angular modes as in (53.17). Equating the terms for each mode ν on both sides gives the conditions

$$\mathrm{e}^{-\mathrm{i}\nu\theta} = \sum_{n=0}^{N-1} g_n(\theta)\, \mathrm{e}^{-\mathrm{i}\nu\theta_n} \quad \forall \nu \,. \tag{53.61}$$

This approach is called *mode matching* [53.27]. Solving (53.61) for the weighting factors yields the panning functions for a single source. A special case is discussed next.

Closed-Form Solution for the Panning Function

For special loudspeaker setups, the panning functions can be determined in closed form [53.27]. As an example, consider a setup of N loudspeakers with equidistant angular spacing

$$\theta_n = n\frac{2\pi}{N} \,, \qquad n = 0, \ldots, N-1 \,. \tag{53.62}$$

For convenience in the following derivation, N is assumed to be odd. A similar result is obtained for an even number of loudspeakers with a slightly different indexing of the loudspeakers (n) and modes (ν).

Now determine the N unknown weighting factors $g_n(\theta)$ for the reproduction of a plane wave from the N equations (53.61)

$$\mathrm{e}^{-\mathrm{i}\nu\theta} = \sum_{n=0}^{N-1} g_n(\theta)\, \mathrm{e}^{-\mathrm{i}\nu n \frac{2\pi}{N}} \,, \quad -N' \le \nu \le N' \tag{53.63}$$

where $N' = (N-1)/2$. Since N is odd, N' is an integer. Taking the sum of the terms on the left side

$$\frac{1}{N} \sum_{\nu=-N'}^{N'} \mathrm{e}^{-\mathrm{i}\nu\theta} \cdot \mathrm{e}^{\mathrm{i}\nu m \frac{2\pi}{N}} \tag{53.64}$$

and in the same way on the right side and considering the orthogonality of the complex exponentials

$$\frac{1}{N} \sum_{\nu=-N'}^{N'} \mathrm{e}^{\mathrm{i}\nu(m-n)\frac{2\pi}{N}} = \begin{cases} 1 & m = n \\ 0 & \text{otherwise} \end{cases} \tag{53.65}$$

gives the weighting factors $g_n(\theta)$ in closed form

$$g_n(\theta) = \frac{1}{N} \sum_{\nu=-N'}^{N'} \mathrm{e}^{\mathrm{i}\nu(\theta-\theta_n)} = \frac{\sin\left[N(\theta-\theta_n)/2\right]}{N \sin\left[(\theta-\theta_n)/2\right]} \,. \tag{53.66}$$

This result can also be obtained by expressing (53.63) as

$$\mathrm{e}^{-\mathrm{i}\nu\theta} = \mathrm{DFT}_N \left\{ g_n(\theta) \right\} \,, \tag{53.67}$$

where DFT_N denotes the discrete Fourier transform of length N. Considering both $\mathrm{e}^{-\mathrm{i}\nu\theta}$ and $g_n(\theta)$ as periodic sequences of length N and taking the inverse of (53.67) gives the same result as (53.66).

Note from (53.66) that the panning function is nonzero not only for two or three but for all N loudspeakers. The panning functions may also be negative

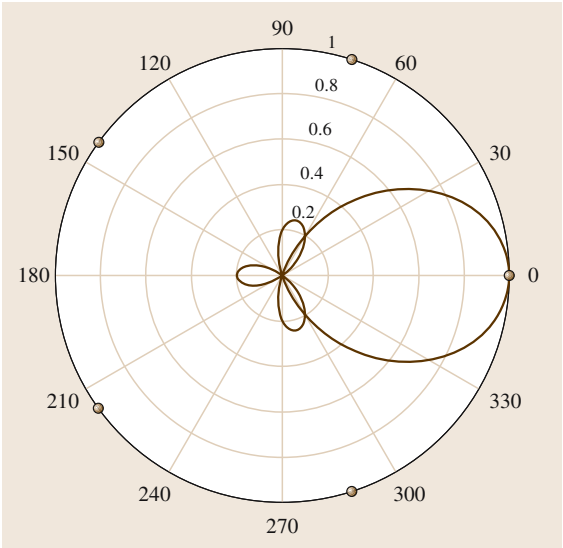


Fig. 53.7 Weighting function $g_0(\theta)$ for the loudspeaker channel $n = 0$ of a two-dimensional ambisonics system with five equally spaced loudspeakers

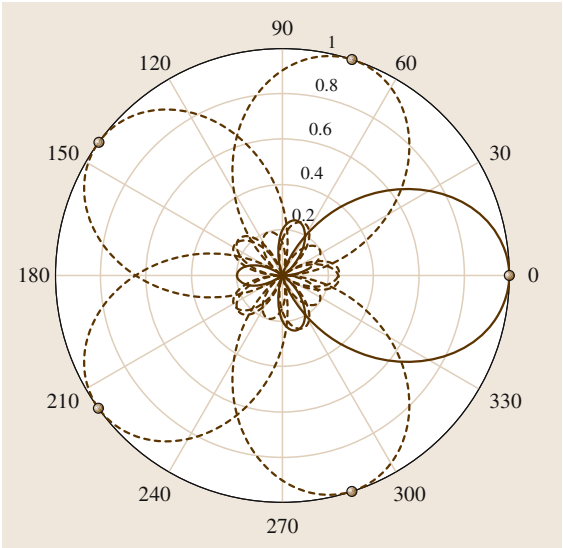


Fig. 53.8 Superposition of the weighting functions for a two-dimensional ambisonics system with five equally spaced loudspeakers

for some loudspeakers. Figure 53.7 shows the weighting factor $g_0(\theta)$ for $N = 5$. The superposition of all weighting factors for $n = 0, \ldots, 4$ is given in Fig. 53.8, which corresponds to Fig. 53.6 for vector-based amplitude panning.

Weighting Functions for General Wave Fields

Now a superposition of plane waves from the directions θ_n is used to represent a general solution of the wave equation

$$P_\mathrm{p}(\boldsymbol{r}, \omega) = \sum_{n=0}^{N-1} P_\mathrm{p}(\boldsymbol{r}, \omega, \theta_n)\,. \tag{53.68}$$

In contrast to (53.58), $P_\mathrm{p}(\boldsymbol{r}, \omega)$ is not constrained to be a plane wave. Still, mode matching can be applied here. Expanding $P_\mathrm{p}(\boldsymbol{r}, \omega)$ as in (53.24) and the plane waves $P_\mathrm{p}(\boldsymbol{r}, \omega, \theta_n)$ as in (53.17) and matching the factors of the modes $\mathrm{e}^{\mathrm{i}\nu\alpha}$ gives

$$G^\circ(\omega, \nu) = \sum_{n=0}^{N-1} G(\omega, \theta_n)\, \mathrm{e}^{\mathrm{i}\nu\theta}\,. \tag{53.69}$$

The angular coefficients $G^\circ(\omega, \nu)$ represent the spatial structure of the sound field to be reproduced. The real and imaginary parts of the corresponding time functions $g^\circ(t, \nu)$ are called *Ambisonics signals*. Their determination is discussed in later. The time functions $g_n(t) = g(t, \theta_n)$ corresponding to $G(\omega, \theta_n)$ are the driving functions of the loudspeakers. Theoretically, they are found by solving (53.69).

However, in practical applications, it is neither possible nor desirable to perform mode matching for a large number modes. If only few loudspeakers are available, then only a few of the Ambisonics signals are required to solve (53.69). If a large number of loudspeakers is available, then the modes of the sound field would have to be known with a sufficient accuracy. Therefore Ambisonics is usually restricted to a low number of modes.

First-Order Ambisonics

The most simple implementation of Ambisonics considers only the terms for $|\nu| \le 1$ of the expansion (53.26)

$$\begin{aligned} P_\mathrm{p}(\boldsymbol{r}, \omega) \approx\; & J_0(kr)\, G^\circ(\omega, 0) \\ & + 2\mathrm{i} J_1(kr)\, G^\circ_\mathrm{R}(\omega, 1) \cos\alpha \\ & - 2\mathrm{i} J_1(kr)\, G^\circ_\mathrm{I}(\omega, 1) \sin\alpha\,, \end{aligned} \tag{53.70}$$

Thus the first-order approximation of the sound field $P_\mathrm{p}(\boldsymbol{r}, \omega)$ can now be represented by the real values $G^\circ(\omega, 0)$, $G^\circ_\mathrm{R}(\omega, 1)$, and $G^\circ_\mathrm{I}(\omega, 1)$.

Second- and Higher-Order Ambisonics

Truncating the series (53.26) for values $|\nu| \le 2$ leads to an expansion similar to (53.70) but with additional terms involving the real and imaginary part of

$G^\circ(\omega, 2)$ and the trigonometric terms $\cos 2\alpha$ and $\sin 2\alpha$. These additional terms provide higher angular resolution but only for larger distances of the origin, since the higher-order Bessel functions have small values for small values of kr. This method is called *second-order Ambisonics*.

The effect of increased angular resolution is more pronounced if terms for $2 < |\nu|$ are added. The corresponding *higher-order Ambisonics* methods are usually combined with a three-dimensional approach discussed in Sect. 53.5.2.

Determination of the Ambisonics Signals

The Ambisonics signals are the inverse Fourier transforms of the angular components $G^\circ(\omega, 0)$, $G^\circ_\mathrm{R}(\omega, \nu)$, and $G^\circ_\mathrm{I}(\omega, \nu)$ for $\nu > 1$. They contain the necessary temporal and spatial information about the sound field subject to the approximation induced by the series truncation discussed in Sect. 53.5.1. There are two principal methods for their determination, a model-based method and a data-based method relying on microphone recordings.

In the model-based method, the angular coefficients $G^\circ(\omega, \nu)$ are determined from a model of the desired sound field through an evaluation of (53.25). For example, modeling the sound field as a superposition of K plane waves from different directions θ_k, $k = 0, \ldots, K-1$ results in

$$G^\circ(\omega, \nu) = \sum_{k=0}^{K-1} G(\omega, \theta_k) \mathrm{e}^{-\mathrm{i}\nu\theta_k} \,. \tag{53.71}$$

This process is called *spatial encoding*.

The data-based method obtains the Ambisonics signals directly from suitable microphone recordings. As an example consider first-order Ambisonics according to (53.70). The terms $G^\circ_\mathrm{R}(\omega, 1)$ and $G^\circ_\mathrm{I}(\omega, 1)$ each correspond to a spatial *figure-of-eight* characteristic of pressure gradient microphones, differing by an angle of 90°. The term $G^\circ(\omega, 0)$ is not associated with any function of α and corresponds to the spatial characteristic of an omnidirectional microphone. Thus the first-order Ambisonics signals for an arbitrary sound field can be recorded by an arrangement of an omnidirectional and two pressure gradient microphones, where the look direction of the latter differs by 90°. The signals delivered by these microphones correspond directly to the sound pressure and the particle velocity components and thus to $G^\circ(\omega, 0)$, $G^\circ_\mathrm{R}(\omega, 1)$, and $G^\circ_\mathrm{I}(\omega, 1)$ according to (53.34) and (53.35).

The data-based method is not easily extended to second- or higher-order Ambisonics, due to the lack of higher-order microphones. Recording approaches with approximations of multipole microphones and with circular microphone arrays are discussed in [53.27–30].

Determination of the Loudspeaker Driving Signals

Ambisonics does not require that the loudspeaker configuration is known when determining the Ambisonics signals. Different loudspeaker configuration can be used for the same set of Ambisonics signals. However, the determination of the loudspeaker driving signals $g_n(t)$ is particularly simple if the loudspeakers are distributed at equal angular spacing, i. e., if the loudspeakers are positioned at $\theta_n = n\,2\pi/N$ for $n = 0, \ldots, N-1$. If the number of Ambisonics signals matches the number of loudspeakers then (53.69) turns into

$$\begin{aligned} G^\circ(\omega, \nu) &= \sum_{n=0}^{N-1} G(\omega, \theta_n) \, \mathrm{e}^{-\mathrm{i}\nu n \frac{2\pi}{N}} \\ &= \mathrm{DFT}_N\{G(\omega, \theta_n)\} \,. \end{aligned} \tag{53.72}$$

This relation is similar to (53.67), but now it is valid for general wave fields and not only for one plane wave. The Fourier transforms $G(\omega, \theta_n)$ of the driving signals then follow from the inverse DFT

$$G(\omega, \theta_n) = \mathrm{DFT}_N^{-1}\{G^\circ(\omega, \nu)\}. \tag{53.73}$$

The determination of the loudspeaker driving signals and the wave field they produce is now discussed in detail for first-order Ambisonics.

First-order Ambisonics for two spatial dimensions uses only the Ambisonics signals $G^\circ(\omega, 0)$, $G^\circ_\mathrm{R}(\omega, 1)$, and $G^\circ_\mathrm{I}(\omega, 1)$ and thus requires at least three loudspeakers for reproduction. A popular choice for simple planar Ambisonics is $N = 4$ [53.16]. In this case, (53.73) becomes

$$\begin{aligned} G(\omega, \theta_n) &= \mathrm{DFT}_4^{-1}\{G^\circ(\omega, \nu)\} \\ &= \frac{1}{4} G^\circ(\omega, 0) + \frac{1}{2} \mathrm{Re}\left\{ G^\circ(\omega, 1) \mathrm{e}^{\mathrm{i}n\frac{2\pi}{N}} \right\} \\ &= \frac{1}{4} G^\circ(\omega, 0) + \frac{1}{2} G^\circ_\mathrm{R}(\omega, 1) \cos\theta_n \\ &\quad - \frac{1}{2} G^\circ_\mathrm{I}(\omega, 1) \sin\theta_n \end{aligned} \tag{53.74}$$

or

$$
\begin{aligned}
G(\omega,\theta_0) &= \frac{1}{4}G^{\circ}(\omega,0)+\frac{1}{2}G^{\circ}_{\mathrm{R}}(\omega,1),\\
G(\omega,\theta_1) &= \frac{1}{4}G^{\circ}(\omega,0)-\frac{1}{2}G^{\circ}_{\mathrm{I}}(\omega,1),\\
G(\omega,\theta_2) &= \frac{1}{4}G^{\circ}(\omega,0)-\frac{1}{2}G^{\circ}_{\mathrm{R}}(\omega,1),\\
G(\omega,\theta_3) &= \frac{1}{4}G^{\circ}(\omega,0)+\frac{1}{2}G^{\circ}_{\mathrm{I}}(\omega,1)\,. \qquad (53.75)
\end{aligned}
$$

These driving signals for the loudspeakers actually reproduce those properties of the original wave field which have been encoded in the Ambisonics signals.

To verify this statement, consider the wave field $Q_{\mathrm{p}}(r,\alpha,\omega)$ that is produced as a superposition of $N=4$ plane waves from the directions $\theta_n = n\,\pi/2$ for $n=0,\ldots,3$. According to (53.19) and (53.22), $Q_{\mathrm{p}}(r,\alpha,\omega)$ is given by

$$
\begin{aligned}
&Q_{\mathrm{p}}(r,\alpha,\omega)\\
&=\sum_{n=0}^{3} G(\omega,\theta_n)\,\hat{p}_{\mathrm{p}}(r,\alpha,\theta_n)\\
&= J_0(kr)\sum_{n=0}^{3} G(\omega,\theta_n)\\
&\quad +2\sum_{\nu=1}^{\infty}\mathrm{i}^{\nu}J_{\nu}(kr)\sum_{n=0}^{3} G(\omega,\theta_n)\cos\left[\nu(\alpha-\theta_n)\right]\,. \qquad (53.76)
\end{aligned}
$$

Now consider the special values $Q_{\mathrm{p}}(0,\alpha,\omega)$ and $\frac{\partial}{\partial r}Q_{\mathrm{p}}(r,\alpha,\omega)\big|_{r=0}$. They follow from (53.76) with (53.27) as

$$
Q_{\mathrm{p}}(0,\alpha,\omega)=\sum_{n=0}^{3} G(\omega,\theta_n) \qquad (53.77)
$$

and

$$
\begin{aligned}
&\frac{\partial}{\partial r}Q_{\mathrm{p}}(r,\alpha,\omega)\bigg|_{r=0}\\
&=\mathrm{i}k\sum_{n=0}^{3} G(\omega,\theta_n)\cos(\alpha-\theta_n) \qquad (53.78)\\
&=\mathrm{i}k[G(\omega,\theta_0)-G(\omega,\theta_2)]\cos\alpha\\
&\quad +\mathrm{i}k[G(\omega,\theta_1)-G(\omega,\theta_3)]\sin\alpha\,.
\end{aligned}
$$

Inserting the loudspeaker signals from (53.75) gives

$$
Q_{\mathrm{p}}(0,\alpha,\omega)=G^{\circ}(\omega,0)=P_{\mathrm{p}}(0,\alpha,\omega) \qquad (53.79)
$$

and

$$
\begin{aligned}
&\frac{\partial}{\partial r}Q_{\mathrm{p}}(r,\alpha,\omega)\bigg|_{r=0}\\
&=\mathrm{i}k\left[G^{\circ}_{\mathrm{R}}(\omega,1)\cos\alpha-G^{\circ}_{\mathrm{I}}(\omega,1)\sin\alpha\right] \qquad (53.80)
\end{aligned}
$$

such that

$$
\begin{aligned}
\frac{\partial}{\partial r}Q_{\mathrm{p}}(r,0,\omega)\bigg|_{r=0} &= \frac{\partial}{\partial r}P_{\mathrm{p}}(r,0,\omega)\bigg|_{r=0}\,, \qquad (53.81)\\
\frac{\partial}{\partial r}Q_{\mathrm{p}}(r,\frac{\pi}{2},\omega)\bigg|_{r=0} &= \frac{\partial}{\partial r}P_{\mathrm{p}}(r,\frac{\pi}{2},\omega)\bigg|_{r=0}\,. \qquad (53.82)
\end{aligned}
$$

Comparing (53.28) and (53.79) and also (53.32) and (53.33) with (53.81) and (53.82) shows that the sound pressure and the components of the particle velocity are correctly reproduced at the position $r=0$.

For loudspeaker arrangements other than equal angular spacing, the system of equations (53.69) cannot be formulated conveniently as a DFT. However it is still possible to solve (53.69) for the loudspeaker signals g_n, either exact or in the least-squares sense depending on the relation between the number of Ambisonics signals and the number of loudspeakers [53.31].

53.5.2 Three-Dimensional Ambisonics

Three-dimensional Ambisonics systems are based on a three-dimensional loudspeaker arrangement, which produces approximately plane waves from arbitrary spatial directions. Three-dimensional Ambisonics is often considered in the context of higher-order Ambisonics [53.31–33]. The mathematical basis is a modal expansion similar to (53.24) and (53.26), where the angular expansion coefficients are defined on a sphere described in terms of an azimuth and an elevation angle. These modal expansion functions are called *spherical harmonics* and are more involved than the two-dimensional expansion from (53.24). Therefore only a short description is given here; for mathematical details see [53.3, 4, 34, 35].

The three-dimensional Ambisonics signals are the coefficients of the expansion into spherical harmonics. Similar to Sect. 53.5.1 they can be obtained by model- and data-based methods.

In the model-based approach, the Ambisonics signals are calculated from the geometrical and acoustical description of a virtual scene [53.36].

The simplest method for the data-based approach consists of an arrangement of four microphones, one omnidirectional microphone, and three velocity microphones at the same spot. The look directions of the

velocity microphones are orthogonal so that they cover all three spatial dimensions. This way only first-order Ambisonics signals can be recorded.

Ambisonics signals of higher order can be recorded with spherical microphone arrays, i. e., an arrangement of pressure microphones distributed evenly over a spherical baffle. Such arrangements can be built as compact as a single microphone and are sometimes called *sound field microphones*. The theory of spherical microphone arrays and applications to spatial sound recording are presented in [53.33, 37–39].

53.5.3 Extensions of Ambisonics

So far two- and three-dimensional Ambisonics has been described as a superposition of plane waves from a finite number of distant loudspeakers. Now, two extensions are briefly discussed.

The sound field of a loudspeaker is modeled more correctly as a spherical wave rather than a plane wave. Taking the curvature of the wavefront into account, not only the direction but also distance of a sound source can be encoded. A higher-order Ambisonics format based also on the near field of a source has been introduced [53.32] under the name near-field-coding higher-order Ambisonics (NFC-HOA).

Even for higher-order Ambisonics, the number of modes is usually rather moderate. A spherical array takes spatial samples of the sound field at the microphone positions. Therefore the number of modes that can be uniquely determined is restricted by spatial aliasing. In practical microphone arrangements, orders of up to five have been reported [53.38].

However, theoretical investigations have shown [53.40, 41] that infinite-order Ambisonics is a particular case of an approach called *holophony* or *acoustic holography*. A technique for an approximate realization of acoustic holography is described in the next section.

53.6 Wave Field Synthesis

Wave field synthesis is a sound reproduction technique that overcomes certain limitations of conventional surround sound methods. It is based on a physical description of the propagation of acoustic waves. Wave field synthesis uses loudspeaker array technology to correctly reproduce sound fields without the *sweet spot* limitation discussed above. Wave field synthesis techniques are formulated in terms of the acoustic wave equation and their Green's functions [53.13, 42, 43]. The following description is an abridged version of a more-detailed account in [53.44].

53.6.1 Description of Acoustical Scenes by the Kirchhoff–Helmholtz Integral

Consider the sound reproduction scenario depicted in Fig. 53.2. The wave field emitted by an arbitrary virtual source $Q(\boldsymbol{z}, \omega)$ shall be reproduced in the bounded region V. This region is called the *listening region*, since the listeners reside there. The virtual source $Q(\boldsymbol{z}, \omega)$ shall lie outside of the listening region V. The limitation to one virtual source poses no constraints on the wave field to be reproduced, since this source may have arbitrary shape and frequency characteristics. Additionally, multiple sources can be reproduced by the principle of superposition.

For this sound reproduction scenario the Green's function $G_{3D}(\boldsymbol{z}|\boldsymbol{z}', \omega)$ and its directional gradient can be understood as the field emitted by sources placed on the boundary ∂V. These sources are called *secondary sources*. The strength of these sources is determined by the pressure $P(\boldsymbol{z}', \omega)$ and the directional pressure gradient $\frac{\partial}{\partial \boldsymbol{n}} P(\boldsymbol{z}', \omega)$ of the virtual source field $Q(\boldsymbol{x}', \omega)$ on ∂V.

Now the Kirchhoff–Helmholtz integral (53.37) can be interpreted as follows: if appropriately chosen secondary sources are driven by the values of the sound pressure and the directional pressure gradient caused by the virtual source $Q(\boldsymbol{x}', \omega)$ on the boundary ∂V, then the wave field within the region V is equivalent to the wave field which would have been produced by the virtual source inside V. Thus, the theoretical basis of sound reproduction is described by the Kirchhoff–Helmholtz integral.

The three-dimensional free-field Green's function is given by (53.36). In the context of sound reproduction it can be interpreted as the field of a monopole point source distribution on the surface ∂V. The Kirchhoff–Helmholtz integral (53.37) also involves the directional gradient of the Green's function. The directional gradi-

ent of the three-dimensional free-field Green's function can be interpreted as the field of a dipole source whose main axis lies in direction of the normal vector $\boldsymbol{n}$. Thus, the Kirchhoff–Helmholtz integral states in this case, that the acoustic pressure inside the volume V can be controlled by a monopole and a dipole point source distribution on the surface ∂V enclosing the volume V.

This interpretation of the Kirchhoff–Helmholtz integral sketches a first draft of a technical system for spatial sound reproduction. In rough terms, such a system would consist of technical approximations of acoustical monopoles and dipoles by appropriate loudspeakers. These loudspeakers cover the surface of a suitably chosen volume around the possible listener positions. They are excited by appropriate driving functions to reproduce the desired sound field inside the volume. However, there remain a number of fundamental questions to be resolved on the way to a technical realization.

53.6.2 Monopole and Dipole Sources

Technical approximations of acoustical monopoles and dipoles consist of loudspeakers with different types of enclosures. A restriction to only one type of sources would be of advantage for a technical realization. For example the exclusive use of monopole sources facilitates a technical solution with small loudspeakers in closed cabinets.

Reconsidering (53.37) shows that the use of monopole and dipole sources allows a very precise reproduction of the desired wave field: it is recreated as $P(z, \omega)$ for all positions inside V and it is zero outside. Such a restriction is usually not required for spatial sound reproduction. As long as the reproduction is correct inside of V, almost arbitrary sound fields outside may be tolerated, as long as their sound intensity is moderate. This situation suggests the following trade-off: the

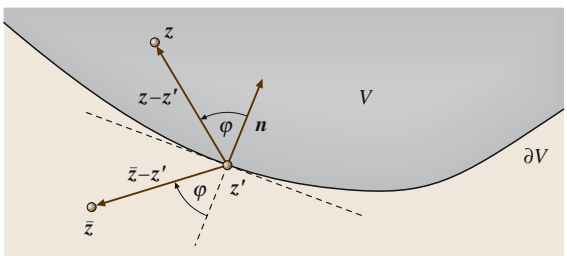


Fig. 53.9 Illustration of the geometry used for the derivation of the modified Green's function for a sound reproduction system using monopole secondary sources only

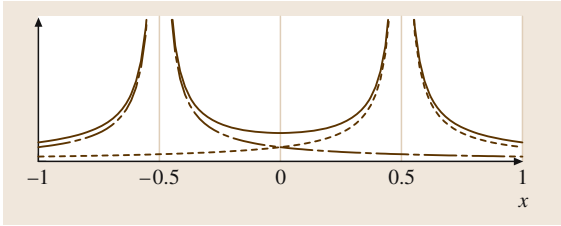


Fig. 53.10 Construction of a Green's function with zero derivative at the boundary $x = 0$ by superposition of the Green's functions of two point sources (only component in x-direction). Free-field source at $x = 0.5$ (*dashed*), mirrored free-field source at $x = -0.5$ (*dash-dotted*), superposition of two sources (*solid*). The *vertical axis* shows the absolute value of the Green's functions in arbitrary units

use of one type of sound sources only and tolerate some sound pressure outside of V.

To realize this trade-off, a Green's function $G_{3D}(z|z', \omega)$ that satisfies boundary conditions of the first or second kind on the surface ∂V is constructed. Since it is desirable to drop the dipoles and to keep the monopole sources, the Green's function of a point source $G^{\mathrm{f}}_{3D}(z|z', \omega)$ according to (53.36) is chosen as the basic building block. Then for each position z' on the boundary, the Green's function $G^{\mathrm{f}}_{3D}(z|z', \omega)$ for a position z inside V and the Green's function $G^{\mathrm{f}}_{3D}(\bar{z}(z)|z', \omega)$ for a position $\bar{z}(z)$ outside V are superposed

$$G_{3D}(z|z', \omega) = G^{\mathrm{f}}_{3D}(z|z', \omega) + G^{\mathrm{f}}_{3D}(\bar{z}(z)|z', \omega)\,. \tag{53.83}$$

The position $\bar{z}(z)$ is chosen as the mirror image of z with respect to the tangent plane in z' on the surface ∂V (see Fig. 53.9). The tangent plane is characterized by the unit vector $\boldsymbol{n}$. The notation $\bar{z}(z)$ indicates that $\bar{z}$ depends on z.

Figure 53.10 exemplifies the construction of $G_{3D}(z|z', \omega)$. It shows a one-dimensional cut in the direction of $\boldsymbol{n}$. The superposition of two free-field Green's functions results in a zero normal derivative of $G_{3D}(z|z', \omega)$ at the boundary (Neumann boundary conditions).

Inserting $G_{3D}(z|z', \omega)$ from (53.83) as the Green's function in the Kirchhoff–Helmholtz integral (53.37) leads to

$$P(z, \omega) = -\oint_{\partial V} G_{3D}(z|z', \omega)\, \frac{\partial}{\partial \boldsymbol{n}} P(z', \omega)\, \mathrm{d}z' \tag{53.84}$$

for $z \in V$. Outside V the wave field consists of a mirrored version of the wave field inside V. An example for a cir-

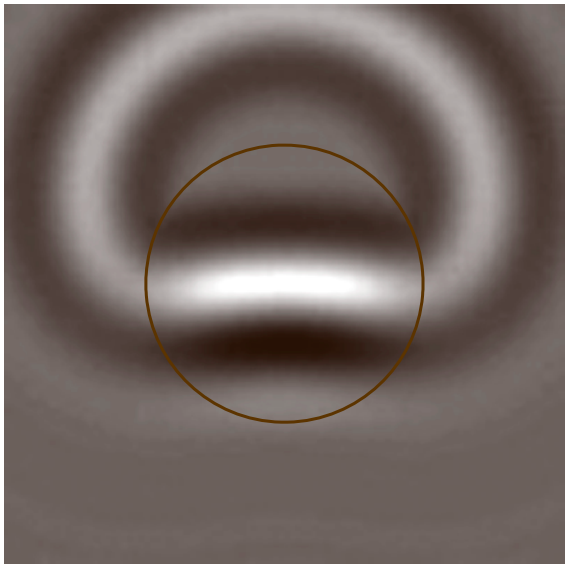


Fig. 53.11 Simulated sound field created by a distribution of monopoles on a *circle* (shown in *black*). Inside the circle: approximation of a plane wave, outside of the circle: mirrored sound field caused by a reflection of the plane wave from inside with respect to the *black circle*

cular boundary is shown in Fig. 53.11. The result (53.84) is also known as the type I Raleigh integral [53.35]. On the boundary ∂V, the following holds:

$$G_{3D}(z|z',\omega) = 2\,G^{f}_{3D}(z|z',\omega)\,. \tag{53.85}$$

The factor of two follows from the superposition shown in Fig. 53.10. Thus the sound field $P(z,\omega)$ inside a volume V can be generated by a distribution of monopole sources on the surface ∂V as

$$P(z,\omega) = -\oint_{\partial V} 2\,G^{f}_{3D}(z|z',\omega)\,\frac{\partial}{\partial \boldsymbol{n}}\,P(z',\omega)\,\mathrm{d}z'\,. \tag{53.86}$$

53.6.3 Reduction to Two Spatial Dimensions

The requirement of creating a distribution of sources on a whole surface around a listening space may be impractical for many sound reproduction systems. This Section shows how to reduce the source distribution from a surface around the listeners to a closed curve in a horizontal plane preferably at the height of the listeners' ears. For convenience this height is denoted by $z = 0$.

The Kirchhoff–Helmholtz integral is now specialized to sound fields that do not depend on the

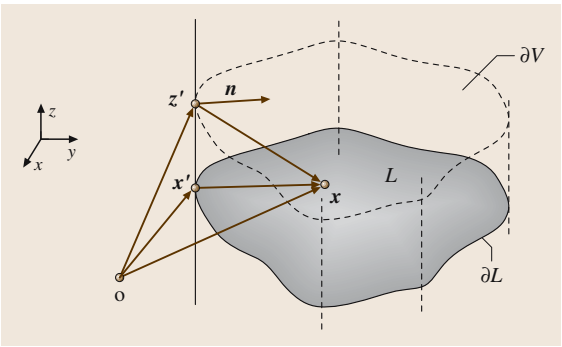


Fig. 53.12 Illustration of a sound field in a prism oriented parallel to the z-axis. For this special geometry, the intersection with any plane parallel to the xy-plane is independent of z. If the source strength on the boundary ∂V is also independent of z, the sound field within the prism does not depend on the z-coordinate

z-coordinate. Then the shape of the volume for the integration in (53.37) turns into a prism oriented parallel to the z-axis (Fig. 53.12). This rather special spatial arrangement allows the transition to a 2-D description of the Kirchhoff–Helmholtz integral.

Since the sound field is assumed to be independent of z, $P(z,\omega)$ depends only on x and y. Furthermore, any vector normal to the surface ∂V has no component in the z-direction and also the normal derivative of $P(z,\omega)$ is independent of z. Thus the surface integration with respect to z' in (53.37) can be split into a contour integration with respect to $\boldsymbol{x}'$ and an integration with respect to z'. The contour ∂L is defined by the intersection of the prism with the xy-plane. This procedure turns the surface integral in (53.86) into

$$\begin{aligned}
&\oint_{\partial V} G^{f}_{3D}(z|z',\omega)\,\frac{\partial}{\partial \boldsymbol{n}}\,P(z',\omega)\,\mathrm{d}z' \\
&= \oint_{\partial L}\int_{-\infty}^{\infty} G^{f}_{3D}(z|z',\omega)\,\frac{\partial}{\partial \boldsymbol{n}}\,P(z',\omega)\,\mathrm{d}z'\,\mathrm{d}\boldsymbol{x}' \\
&= \oint_{\partial L}\left[\int_{-\infty}^{\infty} G^{f}_{3D}(z|z',\omega)\,\mathrm{d}z'\right]\frac{\partial}{\partial \boldsymbol{n}}\,P(z',\omega)\,\mathrm{d}\boldsymbol{x}'\,.
\end{aligned} \tag{53.87}$$

Since $P(z',\omega)$ on the contour ∂L is described by a two-dimensional function $P(z',\omega) = P_c(\boldsymbol{x}',\omega)$, a two-dimensional version of the Raleigh I integral is given by

$$P_c(\boldsymbol{x},\omega) = -\oint_{\partial L} 2G^{f}_{2D}(\boldsymbol{x}|\boldsymbol{x}',\omega)\,\frac{\partial}{\partial \boldsymbol{n}}\,P_c(\boldsymbol{x}',\omega)\,\mathrm{d}\boldsymbol{x}' \tag{53.88}$$

with

$$G^{\mathrm{f}}_{2\mathrm{D}}(\boldsymbol{x}|\boldsymbol{x}',\omega) = \int_{-\infty}^{\infty} G^{\mathrm{f}}_{3\mathrm{D}}(\boldsymbol{x}|z',\omega)\,\mathrm{d}z' \,. \tag{53.89}$$

Evaluating this integral gives [53.9, 3.876]

$$G^{\mathrm{f}}_{2\mathrm{D}}(\boldsymbol{x}|\boldsymbol{x}',\omega) = \tilde{G}^{\mathrm{f}}_{2\mathrm{D}}(\rho,\omega) = -\frac{\mathrm{i}}{4} H_0^{(2)}\Big(\frac{\omega}{c}\rho\Big) \tag{53.90}$$

with the Hankel function of the second kind and order zero [53.9] being

$$H_0^{(2)}(u) = J_0(u) - \mathrm{i}N_0(u) \,, \tag{53.91}$$

where $J_0(u)$ and $N_0(u)$ are, respectively, the Bessel and Neumann functions of the first kind and order zero.

Due to the circular symmetry, $G^{\mathrm{f}}_{2\mathrm{D}}(\boldsymbol{x}|\boldsymbol{x}',\omega)$ depends only on the distance ρ between the listener position $\boldsymbol{x}$ and the line source at $\boldsymbol{x}'$

$$\rho = \|\boldsymbol{x}-\boldsymbol{x}'\| = \sqrt{(x-x')^2+(y-y')^2} \tag{53.92}$$

and can thus be replaced by $\tilde{G}^{\mathrm{f}}_{2\mathrm{D}}(\rho)$ as in (53.90). With the far-field approximation of the Hankel function for large values of ρ [53.8]

$$H_0^{(2)}\Big(\frac{\omega}{c}\rho\Big) \approx \sqrt{\frac{2\mathrm{i}}{\pi\left(\frac{\omega}{c}\rho\right)}}\; \mathrm{e}^{-\mathrm{i}\left(\frac{\omega}{c}\rho\right)} \tag{53.93}$$

it follows that

$$\tilde{G}^{\mathrm{f}}_{2\mathrm{D}}(\rho,\omega) = H(\omega)\,A(\rho)\,G^{\mathrm{f}}_{3\mathrm{D}}(\boldsymbol{x}|\boldsymbol{x}',\omega)\,, \tag{53.94}$$

where

$$H(\omega) = \sqrt{\frac{c}{\mathrm{i}\omega}} = \sqrt{\frac{1}{\mathrm{i}k}} \tag{53.95}$$

causes a spectral shaping and

$$A(\rho) = \sqrt{2\pi\,\rho} \tag{53.96}$$

causes an amplitude modification of $G^{\mathrm{f}}_{2\mathrm{D}}(\boldsymbol{x}|\boldsymbol{x}',\omega)$.

Inserting (53.94) into (53.88) gives, with (53.92),

$$P_{\mathrm{c}}(\boldsymbol{x},\omega) = -\oint_{\partial L} G^{\mathrm{f}}_{3\mathrm{D}}(\boldsymbol{x}|\boldsymbol{x}',\omega)\,D(\boldsymbol{x}|\boldsymbol{x}',\omega)\,\mathrm{d}\boldsymbol{x}' \tag{53.97}$$

for $\boldsymbol{x} \in L$ where

$$D(\boldsymbol{x}|\boldsymbol{x}',\omega) = 2A\big(\|\boldsymbol{x}-\boldsymbol{x}'\|\big)\,H(\omega)\,\frac{\partial}{\partial \boldsymbol{n}} P_{\mathrm{c}}(\boldsymbol{x}',\omega) \,. \tag{53.98}$$

Here, $G^{\mathrm{f}}_{3\mathrm{D}}(\boldsymbol{x}|\boldsymbol{x}',\omega)$ denotes the Green's function of the monopole sources on the contour ∂L in the xy-plane. It describes the wave propagation in 3-D space, however, the receiver locations $\boldsymbol{x}$ (the listeners' ears) are assumed to reside in the xy-plane as well. $D(\boldsymbol{x}|\boldsymbol{x}',\omega)$ denotes the source signal of the monopoles.

53.6.4 Spatial Sampling

The previous sections showed how the rather general statement of the Kirchhoff–Helmholtz integral can be narrowed down to a model for a spatial reproduction system. A hypothetical distribution of monopole and dipole sources on a 2-D surface around the listener has been replaced by a distribution of monopoles on a 1-D contour in a horizontal plane in the height of the listeners' ears.

For a technical solution, this spatially continuous source distribution has to be replaced by an arrangement of a finite number of loudspeakers with a monopole-like source directivity. The resulting wave field is given by a modification of (53.97), where the integral is substituted by a sum over the discrete loudspeaker positions $\boldsymbol{x}'_n$

$$P_{\mathrm{c}}(\boldsymbol{x},\omega) \approx -\sum_n G^{\mathrm{f}}_{3\mathrm{D}}(\boldsymbol{x}|\boldsymbol{x}'_n,\omega)\,D(\boldsymbol{x}|\boldsymbol{x}'_n,\omega)\,\Delta x'_n \,, \tag{53.99}$$

where $\Delta x'_n$ is the length of the spatial increment between the samples $\boldsymbol{x}'_n$ and is not required to be equidistant. Replacing the continuous-space source distribution by a discrete-space arrangement of loudspeakers can be described as a spatial sampling process. The resulting aliasing effects are discussed, e.g., in [53.45, 46].

53.6.5 Determination of the Loudspeaker Driving Signals

The driving signals for the loudspeakers at positions $\boldsymbol{x}'_n$ follow from (53.98) by inverse Fourier transformation with respect to ω. They are independent of the position of the listener except for the amplitude modification $A(\|\boldsymbol{x}-\boldsymbol{x}'\|)$. In practical situations, the amplitude modification is set to a fixed position inside the volume V. Then the loudspeaker signals are completely independent of the listener's position. The result is an approximate correction of the amplitude deviations for different sources with different distances. However, all other components of $D(\boldsymbol{x}|\boldsymbol{x}'_n,\omega)$ do not depend on any information about listeners at all. So especially frequency compensation by $H(\omega)$ and the computation

of $\partial/\partial \boldsymbol{n} P_c(\boldsymbol{x}'_n, \omega)$ can be performed correctly for all positions inside V.

The crucial point in the determination of the loudspeaker signals is the gradient $\frac{\partial}{\partial \boldsymbol{n}} P_c(\boldsymbol{x}'_n, \omega)$ at the loudspeaker positions. A naive approach is to measure the original sound field with properly positioned and oriented velocity microphones. However, those measurements would be only valid for a reproduction with exactly the same number and position of loudspeakers. More-versatile approaches are again model- and data-based methods.

In the model-based approach, the pressure gradient at the loudspeaker positions is determined from a model of the acoustic scene similar to Fig. 53.2. The simplest model is free-field propagation, where the loudspeaker signals follow from delayed and attenuated versions of the source signals. More-elaborate models take the room acoustics of the recording room into account. Details of the model-based determination of the loudspeaker signals are given, e.g., in [53.44, 47].

In the data-based approach, the spatial characteristics of the room acoustics in the recording room are determined by microphone measurements. From these, a set of room impulse responses is derived for use in wave field synthesis reproduction. Appropriate recording techniques are described, e.g., in [53.28, 48, 49].

References

53.1 L.J. Ziomek: *Fundamentals of Acoustic Field Theory and Space–Time Signal Processing* (CRC, Boca Raton 1995)

53.2 A.D. Pierce: *Acoustics. An Introduction to its Physical Principles and Applications* (Acoustical Society of America, Washington 1991)

53.3 P. Filippi, D. Habault, J.-P. Lefebvre, A. Bergassoli: *Acoustics: Basic Physics, Theory and Methods* (Academic, New York 1999)

53.4 E.G. Williams: *Fourier Acoustics: Sound Radiation and Nearfield Acoustical Holography* (Academic, New York 1999)

53.5 R.N. Bracewell: *The Fourier Transform and its Applications* (McGraw-Hill, New York 1978)

53.6 S. Haykin, B. VanVeen: *Signals and Systems* (Wiley, New York 1999)

53.7 B. Girod, R. Rabenstein, A. Stenger: *Signals and Systems* (Wiley, New York 2001)

53.8 M. Abramowitz, I.A. Stegun: *Handbook of Mathematical Functions* (Dover, London 1972)

53.9 I.S. Gradshteyn, I.M. Ryzhik: *Tables of Integrals, Series, and Products* (Academic, New York 1965)

53.10 I.N. Sneddon: *The Use of Integral Transforms* (McGraw-Hill, New York 1972)

53.11 R. Rabenstein, P. Steffen, S. Spors: Representation of two-dimensional wave fields by multidimensional signals, Signal Process. **86**, 1341–1351 (2006)

53.12 P.M. Morse, H. Feshbach: *Methods of Theoretical Physics. Part I* (McGraw-Hill, New York 1953)

53.13 S. Spors: *Active Listening Room Compensation for Spatial Sound Reproduction Systems* (University Erlangen-Nuremberg 2006) Ph.D. thesis http://www.lnt.de/lms/publications

53.14 J. Blauert: *Spatial Hearing: The Psychophysics of Human Sound Localization* (MIT, Cambridge 1996)

53.15 W.B. Snow: Basic principles of stereophonic sound, IRE Trans. Audio **March-April**, 42–53 (1955)

53.16 F. Rumsey: *Spatial Audio* (Focal, Oxford 2001)

53.17 C. Hugonnet, P. Walder: *Stereophonic Sound Recording* (Wiley, New York 1998)

53.18 D. Griesinger: Stereo and surround panning in practice, 112th Convention (AES 2002)

53.19 V. Pulkki: Virtual sound source positioning using vector base amplitude panning, J. Audio Eng. Soc. **45**(6), 456–466 (1997)

53.20 V. Pulkki: Uniform spreading of amplitude panned virtual sources, Proc. Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE, 1999)

53.21 V. Pulkki, M. Karjalainen, J. Huopaniemi: Analyzing sound source attributes using a binaural auditory model, J. Audio Eng. Soc. **47**(4), 203–217 (1999)

53.22 V. Pulkki, M. Karjalainen: Localization of amplitude-panned virtual sources I: Stereophonic panning, J. Audio Eng. Soc. **49**(9), 739–752 (2001)

53.23 V. Pulkki: Localization of amplitude-panned virtual sources II: Two- and three-dimensional panning, J. Audio Eng. Soc. **49**(9), 753–767 (2001)

53.24 M. Gerzon: Periphony: With-height sound reproduction, J. Audio Eng. Soc. **21**, 2–10 (1973)

53.25 J. Daniel: *Représentation de champs acoustiques, application à la transmission et à la reproduction de scènes sonores complexes dans un contexte multimédia*, Ph.D. thesis, Université Paris 6 (2000) http://gyronymo.free.fr/audio3D/download_Thesis_PwPt.html

53.26 F. Hollerweger: Periphonic sound spatialization in multi-user virtual environments, M.S. thesis, University of Music and Dramatic Arts Graz (2005) http://iem.at/projekte/acoustic/awt/periphonic/periphonic.pdf

53.27 M.A. Poletti: A unified theory of horizontal holographic sound systems, J. Audio Eng. Soc. **48**(12), 1155–1182 (2000)

53.28 E. Hulsebos, T. Schuurmanns, D. de Vries, R. Boone: Circular microphone array recording for discrete

multichannel audio recording, 114th AES Convention (AES, Amsterdam 2003)
53.29 H. Teutsch, W. Kellermann: On the estimation of the number of acoustic wideband sources using eigen-beam processing for circular apertures, Workshop on Applications of Signal Processingto Audio and Acoustics (IEEE, 2005)
53.30 H. Teutsch: *Modal Array Signal Processing: Principles and Applications of Acoustic Wavefield Decomposition* (Springer, Berlin 2007)
53.31 J. Daniel, R. Nicol, S. Moreau: Further investigations of high order Ambisonics and wave field synthesis for holophonic sound imaging, 114th AES Convention (AES, 2003), Convention Paper 5788
53.32 J. Daniel: Spatial sound encoding including near field effect: Introducing distance coding filters and a viable, new Ambisonics format, 23rd International Conference (AES, 2003)
53.33 M.A. Poletti: Three-dimensional surround sound systems based on spherical harmonics, J. Audio Eng. Soc. **11**(11), 1004–1025 (2005)
53.34 A.D. Pierce: Mathematical theory of wave propagation. In: *Handbook of Acoustics*, ed. by M.J. Crocker (Wiley, New York 1998)
53.35 N.A. Gumerov, R. Duraiswami: *Fast Multipole Methods for the Helmholtz Equation in three Dimensions* (Elsevier, Amsterdam 2004)
53.36 T. Betlehem, T.D. Abhayapala: Theory and design of sound field reproduction in reverberant rooms, J. Acoust. Soc. Am. **117**(4), 2100–2111 (2005)
53.37 T.D. Abhayapala, D.B. Ward: Theory and design of high order sound field microphones using spherical micrphone array, Proc. ICASSP (2002)
53.38 J. Meyer, G.W. Elko: Spherical microphone arrays for 3D sound recording. In: *Audio Signal Processing For Next-Generation Multimedia Communication Systems*, ed. by Y. Huang, J. Benesty (Kluwer Academic, Dordrecht 2004) pp. 67–89
53.39 B. Rafaely: Analysis and design of spherical microphone arrays, IEEE Trans. Speech Audio Process. **13**(1), 135–143 (2005)
53.40 R. Nicol, M. Emerit: Reproducing 3D-sound for videoconferencing: A comparison between holophony and ambisonic, Proc. Digital Audio Effects (DAFx) (1998) pp. 17–20
53.41 R. Nicol, M. Emerit: 3D-sound reproduction over an extensive listening area: A hybrid method derived from holophony and ambisonic, 16th International Conference (AES, 1999)
53.42 A.J. Berkhout: A holographic approach to acoustic control, J. Audio Eng. Soc. **36**, 977–995 (1988)
53.43 D. de Vries, E.W. Start, V.G. Valstar: The Wave Field Synthesis concept applied to sound reinforcement: Restrictions and solutions, 96th AES Convention (AES, Amsterdam 1994)
53.44 R. Rabenstein, S. Spors, P. Steffen: Wave field synthesis techniques for spatial sound reproduction. In: *Topics in Acoustic Echo and Noise Control*, ed. by E. Hänsler, G. Schmidt (Springer, Berlin, Heidelberg 2006) pp. 517–545
53.45 S. Spors: Spatial aliasing artefacts produced by linear loudspeaker arrays used for wave field synthesis, Proc. 2nd International Symposium on Communications, Control and Signal Processing (ISCCSP'06) (Marrakech, 2006)
53.46 S. Spors, R. Rabenstein: Spatial aliasing artifacts produced by linear and circular loudspeaker arrays used for wave field synthesis, 120th Convention (AES, Paris 2006)
53.47 A.J. Berkhout, D. de Vries: Acoustic holography for sound control, 86th Convention (AES, 1989), Preprint 2801 (F-3)
53.48 E. Hulsebos, D. de Vries, E. Bourdillat: Improved microphone array configurations for auralization of sound fields by Wave Field Synthesis, J. AES **50**(10), 779–790 (2002)
53.49 E. Hulsebos, D. de Vries: Parameterization and reproduction of concert hall acoustics with a circular microphone array, 112th AES Convention (AES, Munich 2002)

51. Time Delay Estimation and Source Localization

Y. Huang, J. Benesty, J. Chen

A fundamental requirement of microphone arrays is the capability of instantaneously locating and continuously tracking a speech sound source. The problem is challenging in practice due to the fact that speech is a nonstationary random process with a wideband spectrum, and because of the simultaneous presence of noise, room reverberation, and other interfering speech sources. This Chapter presents an overview of the research and development on this technology in the last three decades. Focusing on a two-stage framework for speech source localization, we survey and analyze the state-of-the-art time delay estimation (TDE) and source localization algorithms.

This chapter is organized into two sections. In Sect. 51.2, we will study the TDE problem and review a number of cutting-edge TDE algorithms, ranging from the generalized cross-correlation methods to blind multichannel-identification-based algorithms and the second-order statistics-based multichannel cross-correlation coefficient method to the higher-order statistics-based entropy-minimization approach. In Sect. 51.3, we will investigate the source localization problem from the perspective of estimation theory. The emphasis is on least-squares estimators with closed-form estimates. The spherical intersection, spherical interpolation, and linear-correction spherical interpolation algorithms will be presented.

51.1 Technology Taxonomy

Thanks to the explosive growth in computing power and the continuous drop in the cost and size of acoustic electric transducers, a growing number of speech processing and communication systems can afford to employ multiple microphones. There are two major application areas that are driving the interest in microphone array technologies:

- distant speech acquisition that permits good-quality sound pickup due to the capability to mitigate the effects of room reverberation, ambient noise, and other interfering sound sources; and
- acoustic surveillance and monitoring where sound caused by the events of particular interest is detected, localized, and tracked.

In both applications, there is a primary need to localize a speech source by analyzing the acoustic signals captured by various microphones placed at different positions. A good example of the former is microphone

array beam steering [51.1–4] and a good example of the latter is automatic camera pointing for videoconferencing [51.5–8].

The problem of locating radiative point sources using sensor arrays has long been of great research interest given its theoretical as well as practical importance in a great variety of applications, e.g., radar [51.9, 10], underwater sonar [51.11], and seismology [51.12]. In these applications, source localization is more commonly referred to as direction of arrival (DOA) estimation. A class of celebrated approaches is based on high-resolution spectral analysis, including Capon's minimum variance (MV) spectral estimation method [51.13], and eigenanalysis-based techniques, such as the popular multiple signal classification (MUSIC) algorithm [51.14]. These methods perform statistical fit for DOA with respect to a spatiospectral correlation matrix derived from the signals received at the sensors. By assumption, the source signal needs to be statistically stationary and narrowband, and the source is located in the far field of the sensor array. However, these basic premises are barely met by speech sources. Moreover, the multipath effect is not taken into account in the formulation of DOA estimation. Therefore, the high-resolution DOA estimation algorithms were found to be incompetent for speech source localization, particularly in reverberant acoustic environments.

For speech source localization, microphone array beam scanning and time delay estimation (TDE)-based localization methods are the two most widely used approaches. A beamformer is a spatial filter that operates on the microphone outputs in order to enhance the signal coming from one direction while suppressing noise and interference from other directions. Therefore, steering a microphone array beamformer and scanning across a room for the highest-energy output leads to an estimate of the direction of an active speech source. Using two arrays and by intersecting the two corresponding direction estimates, the source location is determined [51.15, 16]. While the beam scanning method has the remarkable advantage that it can easily be extended to the case of simultaneously localizing multiple speech sources [51.17], it has some intrinsic drawbacks. Since speech is a typical broadband signal, ideally a broadband beamformer whose beam pattern would be the same across the entire speech spectrum needs to be developed, which is a technically challenging problem. In addition, most beamforming algorithms assume that the speech sound source is in the far field in order to make the design problem analytically tractable. When the speech source is close to the microphone array, performance degradation can be expected. On the contrary, if the speech source is in the far field, the localization resolution may not be satisfactory as the source direction is only selected from a set of discrete beam scanning angles. The number of beam scanning angles is therefore a trade-off between resolution and computational complexity.

Alternatively, a TDE-based source localization algorithm involves a two-step procedure [51.18]. In the first step, a set of relative time differences of arrival (TDOAs) for different microphone pairs are calculated. In the second step, the acoustic source location is estimated from the TDOAs and using the a priori knowledge about the locations of the microphones. TDE-based source localization algorithms have many merits: they cope well with both narrowband and broadband source signals; their localization resolution can be flexibly adjusted by varying the sampling rate and microphone array size; the effect of room reverberation is only of concern in the first-step TDE and several recently developed robust TDE algorithms are promising for future practical use; and in general they are computationally efficient. Therefore, after continuous investigation over the last two decades, TDE-based speech source localization schemes have become the technique of choice, especially in recent digital systems. In this chapter, an overview of the state of the art of this class of source localization methods is presented.

51.2 Time Delay Estimation

51.2.1 Problem Formulation and Signal Models

Suppose that there is one speech sound source and N microphones. The TDE problem is concerned with the computation of the relative time difference of arrival (TDOA) between different microphone signals.

Depending on the surrounding acoustic environment, there are two signal models for the TDE problem: the ideal free-field model and the real reverberant model. The former assumes no room reverberation while the latter uses an acoustic channel impulse response, usually a finite impulse response (FIR) filter, to describe the effect of room reverberation.

Ideal Free-Field Model

In an anechoic open space, as shown in Fig. 51.1a, the speech source signal $s(k)$ propagates radiatively and the sound level falls off as a function of distance from the source. Then the signal captured by the n-th microphone at time k can be expressed as follows:

$$x_n(k) = \alpha_n s(k-\tau_n) + b_n(k), \quad n = 1, 2, \cdots, N, \tag{51.1}$$

where α_n $(0 \le \alpha_n \le 1)$ is an attenuation factor due to propagation loss, τ_n is the propagation time, and $b_n(k)$ is additive noise. The time difference of arrival (TDOA) between the i-th and j-th microphones is defined as,

$$\tau_{ij} \triangleq \tau_i - \tau_j, \quad i, j = 1, 2, \cdots, N. \tag{51.2}$$

The noise signal $b_n(k)$ is presumed to be a zero-mean, white, Gaussian random process, and is uncorrelated with the source signal as well as the noise signals at other microphones.

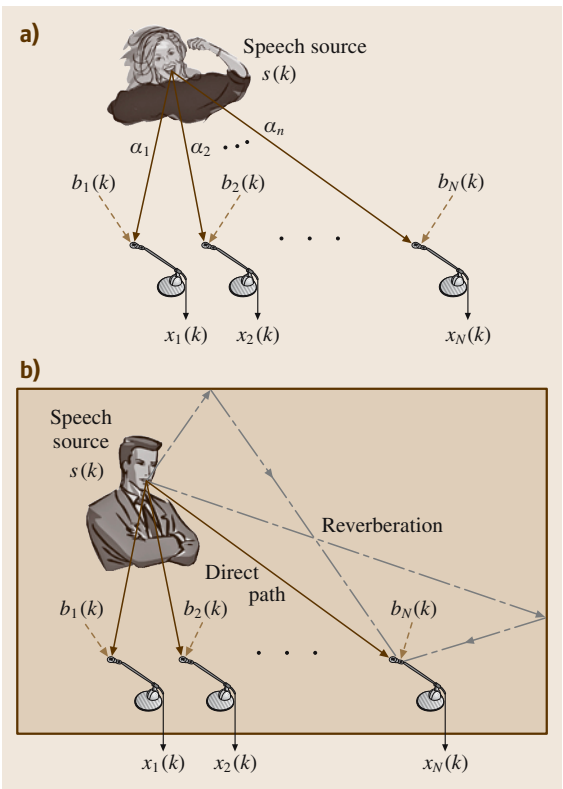


Fig. 51.1a,b Illustration of the two acoustic signal models for time delay estimation: **(a)**ideal free-field model and **(b)**real reverberant model

Real Reverberant Model

While the ideal free-field model has the merit of being simple, it does not take into account room reverberation. Therefore, in a real reverberant environment, the ideal free-field model is inadequate and problematic, and we need a more-comprehensive and more-informative alternative to describe the effect of multipath propagation. The real reverberant model treats the acoustic impulse response with an FIR filter, as illustrated by Fig. 51.1b. In such a single-input multiple-output (SIMO) system, the n-th microphone signal is given by

$$x_n(k) = h_n * s(k) + b_n(k), \quad n = 1, 2, \cdots, N, \tag{51.3}$$

where h_n is the n-th channel impulse response, and the symbol $*$ denotes the linear convolution operator. In vector/matrix form, (51.3) can be rewritten as

$$\boldsymbol{x}_n(k) = \boldsymbol{H}_n \cdot \boldsymbol{s}(k) + \boldsymbol{b}_n(k), \quad n = 1, 2, \cdots, N, \tag{51.4}$$

where

$$\boldsymbol{x}_n(k) = \left[x_n(k) \ \cdots \ x_n(k-L+1) \right]^{\mathrm{T}},$$

$$\boldsymbol{H}_n = \begin{pmatrix} h_{n,0} & \cdots & h_{n,L-1} & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & h_{n,0} & \cdots & h_{n,L-1} \end{pmatrix},$$

$$\boldsymbol{s}(k) = \begin{bmatrix} s(k) & s(k-1) & \cdots & s(k-L+1) \\ \cdots & s(k-2L+2) & \end{bmatrix}^{\mathrm{T}},$$

$$\boldsymbol{b}_n(k) = \left[b_n(k) \ \cdots \ b_n(k-L+1) \right]^{\mathrm{T}},$$

where $[\cdot]^{\mathrm{T}}$ denotes the transpose of a vector or a matrix, and L is the length of the longest channel impulse response in this SIMO system.

In contrast to the ideal free-field model, the time delay τ_n in the real reverberant model is an implicit or hidden parameter. Using such a model, TDE can be performed only after the SIMO system is *blindly* identified (since the source signal is unknown), which looks like a more-difficult problem but is fortunately not insurmountable.

51.2.2 The Family of the Generalized Cross-Correlation Methods

The generalized cross-correlation (GCC) algorithm proposed by *Knapp* and *Carter* [51.19] is so far the most widely used approach to TDE. It employs the ideal free-field model (51.1) and considers only two microphones,

i. e., $N = 2$. Then the TDOA estimate between the two microphones is obtained as the time lag that maximizes the cross-correlation between the filtered signals of the microphone outputs:

$$\hat{\tau}_{12}^{\mathrm{GCC}} = \arg\max_{\tau} r_{x_1x_2}^{\mathrm{GCC}}(\tau)\,, \tag{51.5}$$

where

$$\begin{aligned} r_{x_1x_2}^{\mathrm{GCC}}(\tau) &\triangleq \mathcal{F}^{-1}\left\{\Psi_{x_1x_2}(f)\right\} \\ &= \int_{-\infty}^{\infty} \Psi_{x_1x_2}(f)\mathrm{e}^{\mathrm{i}2\pi f\tau}\,\mathrm{d}f \\ &= \int_{-\infty}^{\infty} \Phi(f)S_{x_1x_2}(f)\mathrm{e}^{\mathrm{i}2\pi f\tau}\,\mathrm{d}f \end{aligned} \tag{51.6}$$

is the GCC function, $\mathcal{F}^{-1}\{\cdot\}$ stands for the inverse discrete-time Fourier transform (IDTFT),

$$S_{x_1x_2}(f) \triangleq E\left\{X_1(f)X_2^*(f)\right\} \tag{51.7}$$

is the cross spectrum, $\Phi(f)$ is a frequency-domain weighting function,

$$\Psi_{x_1x_2}(f) = \Phi(f)S_{x_1x_2}(f) \tag{51.8}$$

is the generalized cross spectrum,

$$X_n(f) = \sum_k x_n(k)\mathrm{e}^{-\mathrm{i}2\pi fk}, \quad n = 1, 2\,,$$

where $E\{\cdot\}$ denotes mathematical expectation, and $(\cdot)^*$ denotes complex conjugate.

There are many different choices of the frequency-domain weighting function $\Phi(f)$, leading to a variety of different GCC methods for TDE. In the following, several most known and practically most useful algorithms in this family will be discussed.

Classical Cross-Correlation

The classical cross-correlation (CCC) method takes $\Phi(f) = 1$. We know from the ideal free-field model (51.1) that

$$X_n(f) = \alpha_n\mathrm{e}^{-\mathrm{i}2\pi f\tau_n}S(f) + B_n(f), \quad n = 1, 2\,. \tag{51.9}$$

Substituting (51.9) into (51.8) and noting that the noise signal at one microphone is uncorrelated with the source signal and the noise signal at the other microphone by assumption, we have

$$\Psi_{x_1x_2}^{\mathrm{CCC}}(f) = \alpha_1\alpha_2\mathrm{e}^{-\mathrm{i}2\pi f\tau_{12}}E\left\{|S(f)|^2\right\}\,. \tag{51.10}$$

The fact that $\Psi_{x_1x_2}^{\mathrm{CCC}}(f)$ depends on the source signal is detrimental for TDE since speech is inherently nonstationary.

Smoothed Coherence Transform

In order to overcome the impact of fluctuating levels of the speech source signal on TDE, an effective way is to pre-whiten the microphone outputs before their cross spectrum is computed. This is equivalent to choosing

$$\Phi(f) = \frac{1}{\sqrt{E\left\{|X_1(f)|^2\right\}E\left\{|X_2(f)|^2\right\}}}\,, \tag{51.11}$$

which leads to the so-called smoothed coherence transform (SCOT) method [51.20]. Substituting (51.9) and (51.11) into (51.8) produces the SCOT cross-spectrum:

$$\begin{aligned} \Psi_{x_1x_2}^{\mathrm{SCOT}}(f) &= \frac{\alpha_1\alpha_2\mathrm{e}^{-\mathrm{i}2\pi f\tau_{12}}E\left\{|S(f)|^2\right\}}{\sqrt{E\left\{|X_1(f)|^2\right\}E\left\{|X_2(f)|^2\right\}}} \\ &= \frac{\alpha_1\alpha_2\mathrm{e}^{-\mathrm{i}2\pi f\tau_{12}}E\{|S(f)|^2\}}{\sqrt{\alpha_1^2E\{|S(f)|^2\}+\sigma_{b_1}^2(f)}\cdot} \\ &\quad\times\frac{1}{\sqrt{\alpha_2^2E\{|S(f)|^2\}+\sigma_{b_2}^2(f)}} \\ &= \frac{\mathrm{e}^{-\mathrm{i}2\pi f\tau_{12}}}{\sqrt{1+\frac{1}{\mathrm{SNR}_1(f)}}\cdot\sqrt{1+\frac{1}{\mathrm{SNR}_2(f)}}}\,, \end{aligned} \tag{51.12}$$

where

$$\begin{aligned} \sigma_{b_n}^2(f) &= E\left\{|B_n(f)|^2\right\}, \\ \mathrm{SNR}_n(f) &= \frac{\alpha_nE\{|S(f)|^2\}}{E\left\{|B_n(f)|^2\right\}}, \quad n = 1, 2\,. \end{aligned}$$

If the signal-to-noise ratios (SNRs) are the same at the two microphones, then we get

$$\Psi_{x_1x_2}^{\mathrm{SCOT}}(f) = \left(\frac{\mathrm{SNR}(f)}{1+\mathrm{SNR}(f)}\right)\cdot\mathrm{e}^{-\mathrm{i}2\pi f\tau_{12}}. \tag{51.13}$$

Therefore, the performance of the SCOT algorithm for TDE would vary with the SNR. But when the SNR is large enough,

$$\Psi_{x_1x_2}^{\mathrm{SCOT}}(f) \approx \mathrm{e}^{-\mathrm{i}2\pi f\tau_{12}}\,,$$

which implies independent TDE performance of the power of the source signal. The SCOT method is theoretically superior to the CCC method only when the noise level is low.

Phase Transform

It becomes clear by examining (51.6) that the TDOA information is conveyed in the phase rather than the amplitude of the cross-spectrum. Therefore, we can simply discard the amplitude and only keep the phase. By setting

$$\Phi(f) = \frac{1}{\left|S_{x_1x_2}(f)\right|}\,, \tag{51.14}$$

we get the phase transform (PHAT) method [51.19]. In this case, the generalized cross spectrum is given by

$$\Psi_{x_1x_2}^{\mathrm{PHAT}}(f) = \mathrm{e}^{-\mathrm{i}2\pi f\tau_{12}}, \tag{51.15}$$

which depends only on the TDOA τ_{12}. Substituting (51.15) into (51.6), we obtain an ideal GCC function:

$$r_{x_1x_2}^{\mathrm{PHAT}}(\tau) = \int_{-\infty}^{\infty} \mathrm{e}^{\mathrm{i}2\pi f(\tau-\tau_{12})}\,\mathrm{d}f = \begin{cases} \infty, & \tau = \tau_{12}, \\ 0, & \text{otherwise.} \end{cases} \tag{51.16}$$

As a result, the PHAT method in general performs better than the CCC and SCOT methods for TDE with respect to a speech sound source.

Discussion: The GCC methods are computationally efficient. They induce very short decision delays and hence have good tracking capability: an estimate is produced almost instantaneously. The GCC methods have been well studied and are found to perform fairly well in moderately noisy and nonreverberant environments [51.21, 22]. In order to improve their robustness to additive noise, many amendments have been proposed [51.6, 23–25]. However, these methods still tend to break down when room reverberation is high. This is insightfully explained by the fact that the GCC methods model the surrounding acoustic environment as an ideal free field and thus have a fundamental weakness in their inability to cope with room reverberation. Cepstral prefiltering was suggested to overcome the multipath propagation effect of room reverberation in [51.26]. However, the improvement is limited and shortcomings still remain. Therefore, the main focus of current research in TDE is on combating the effect of room reverberation.

51.2.3 Adaptive Eigenvalue Decomposition Algorithm

The adaptive eigenvalue decomposition (AED) algorithm [51.27] approaches the TDE problem from a different point of view from the traditional GCC methods. While the AED also considers only two microphones, it adopts the real reverberant rather than the ideal free-field model. It first identifies the two channel impulse responses, and then measures the time difference between the two recognized direct paths as the TDOA estimate. Since the source signal is unknown, the channel identification has to be a blind method.

Following the real reverberant model (51.3) and the fact that, in the absence of additive noise,

$$x_1(k) * h_2 = s(k) * h_1 * h_2 = x_2(k) * h_1\,, \tag{51.17}$$

we deduce the following cross relation in vector/matrix form at time k:

$$\boldsymbol{x}^{\mathrm{T}}(k)\boldsymbol{u} = \boldsymbol{x}_1^{\mathrm{T}}(k)\boldsymbol{h}_2 - \boldsymbol{x}_2^{\mathrm{T}}(k)\boldsymbol{h}_1 = 0, \tag{51.18}$$

where

$$\begin{aligned} \boldsymbol{x}(k) &= \left[\boldsymbol{x}_1^{\mathrm{T}}(k)\;\; \boldsymbol{x}_2^{\mathrm{T}}(k)\right]^{\mathrm{T}}, \\ \boldsymbol{u} &= \left[\boldsymbol{h}_2^{\mathrm{T}}\;\; -\boldsymbol{h}_1^{\mathrm{T}}\right]^{\mathrm{T}}, \\ \boldsymbol{h}_n &= \left[h_{n,0}\;\; h_{n,1}\;\; \cdots\;\; h_{n,L-1}\right]^{\mathrm{T}}, \quad n = 1, 2\,. \end{aligned}$$

Multiplying (51.18) by $\boldsymbol{x}(k)$ from the left-hand side and taking the expectation yields

$$\boldsymbol{R}_{xx}\boldsymbol{u} = \boldsymbol{0}\,, \tag{51.19}$$

where $\boldsymbol{R}_{xx} \triangleq E\left\{\boldsymbol{x}(k)\boldsymbol{x}^{\mathrm{T}}(k)\right\}$ is the covariance matrix of the two microphone signals. This indicates that the vector $\boldsymbol{u}$, which consists of the two impulse responses, is in the null space of $\boldsymbol{R}$. More specifically, $\boldsymbol{u}$ is the eigenvector of $\boldsymbol{R}_x$ corresponding to its eigenvalue 0. If $\boldsymbol{R}_x$ is rank deficient by 1, $\boldsymbol{u}$ can be uniquely determined up to a scale, which is equivalent to saying that the two-channel SIMO system can be blindly identified. Using what has been proved in [51.28], we know that such a two-channel acoustic SIMO system is blindly identifiable using only the second-order statistics (SOS) of the microphone outputs if and only if the following two conditions hold:

- The polynomials formed from $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$ are coprime , i. e., their channel transfer functions share no common zeros ,
- The autocorrelation matrix $\boldsymbol{R}_{ss} = E\{\boldsymbol{s}(k)\boldsymbol{s}^{\mathrm{T}}(k)\}$ of the source signal is of full rank (such that the SIMO system can be fully excited).

In practice, noise always exists and the covariance matrix $\boldsymbol{R}_{xx}$ is positively definite rather than positively semidefinite. As a consequence, $\boldsymbol{u}$ is found as the normalized eigenvector of $\boldsymbol{R}_{xx}$ corresponding to its smallest eigenvalue:

$$\hat{\boldsymbol{u}} = \arg\min_{\boldsymbol{u}} \boldsymbol{u}^{\mathrm{T}}\boldsymbol{R}_{xx}\boldsymbol{u}, \quad \text{subject to } \|\boldsymbol{u}\|^2 = 1\,. \tag{51.20}$$

In the AED algorithm, solving (51.20) is carried out adaptively using a constrained LMS algorithm:

Initialize

$$\hat{\boldsymbol{h}}_n(0) = \left[\tfrac{\sqrt{2}}{2} \;\; 0 \;\cdots\; 0\right]^{\mathrm{T}}, \quad n = 1, 2\,;$$

$$\hat{\boldsymbol{u}}(0) = \left[\hat{\boldsymbol{h}}_2^{\mathrm{T}}(0) \;\; -\hat{\boldsymbol{h}}_1^{\mathrm{T}}(0)\right]^{\mathrm{T}}\,;$$

Compute, $\quad k = 0, 1, \cdots$

$$e(k) = \hat{\boldsymbol{u}}^{\mathrm{T}}(k)\boldsymbol{x}(k),$$

$$\hat{\boldsymbol{u}}(k+1) = \frac{\hat{\boldsymbol{u}}(k) - \mu e(k)\boldsymbol{x}(k)}{\|\hat{\boldsymbol{u}}(k) - \mu e(k)\boldsymbol{x}(k)\|}\,, \tag{51.21}$$

where the adaptation step size μ is a small positive constant.

After the AED algorithm converges, the time difference between the direct paths of the two identified channel impulse responses $\hat{\boldsymbol{h}}_1$ and $\hat{\boldsymbol{h}}_2$ is measured as the TDOA estimate:

$$\hat{\tau}_{12}^{\mathrm{AED}} = \arg\max_l |\hat{\boldsymbol{h}}_{1,l}| - \arg\max_l |\hat{\boldsymbol{h}}_{2,l}|\,. \tag{51.22}$$

51.2.4 Adaptive Blind Multichannel Identification Based Methods

The AED algorithm provides us with a new way to look at the TDE problem, which was found to be particularly useful to combat room reverberation. It applies the more-realistic real-reverberant model to a two-microphone acoustic system at a time and attempts to blindly identify the two-channel impulse responses, from which the embedded TDOA information of interest is then extracted. Clearly the blind two-channel identification technique plays a central role in such an approach. The more accurately the two impulse responses are blindly estimated, the more precisely the TDOA can be inferred. However, for a two-channel system, the zeros of the two channels can be close, especially when their impulse responses are long, which leads to an ill-conditioned system that is difficult to identify. If they share some common zeros, the system becomes unidentifiable (using only second-order statistics) and the AED algorithm may completely fail. It was suggested in [51.29] that this problem can be alleviated by employing more microphones. When more microphones are employed, it is less likely that all channels will share a common zero. As such, blind identification deals with a more well-conditioned SIMO system and the solutions can be globally optimized over all channels. The resulting algorithm is referred to as the adaptive blind multichannel identification (ABMCI)-based TDE algorithm.

The generalization of blind SIMO identification from two channels to multiple (> 2) channels is not straightforward and in [51.30] a systematic way was proposed. Consider a SIMO system with N channels whose outputs are described by (51.3). Each pair of the system outputs has a cross-relation similar to (51.18) in the absence of noise:

$$\boldsymbol{x}_i^{\mathrm{T}}(k)\boldsymbol{h}_j = \boldsymbol{x}_j^{\mathrm{T}}(k)\boldsymbol{h}_i\,, \quad i, j = 1, 2, \cdots, N\,. \tag{51.23}$$

When noise is present or the channel impulse responses are improperly modeled, the cross-relation does not hold and the a priori error signal can be defined as

$$e_{ij}(k+1) = \frac{\boldsymbol{x}_i^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_j(k) - \boldsymbol{x}_j^{\mathrm{T}}(k+1)\hat{\boldsymbol{h}}_i(k)}{\|\hat{\boldsymbol{h}}(k)\|}\,, \quad i, j = 1, 2, \cdots, N\,, \tag{51.24}$$

where $\hat{\boldsymbol{h}}_i(k)$ is the model filter for the i-th channel at time k and

$$\hat{\boldsymbol{h}}(k) = \left[\hat{\boldsymbol{h}}_1^{\mathrm{T}}(k) \;\; \hat{\boldsymbol{h}}_2^{\mathrm{T}}(k) \;\cdots\; \hat{\boldsymbol{h}}_N^{\mathrm{T}}(k)\right]^{\mathrm{T}}\,.$$

The model filters are normalized to avoid a trivial solution whose elements are all zeros. Based on the error signal defined here, a cost function at time $k+1$ is given by

$$J(k+1) = \sum_{i=1}^{N-1}\sum_{j=i+1}^{N} \mathrm{e}_{ij}^2(k+1)\,. \tag{51.25}$$

The multichannel LMS (MCLMS) algorithm updates the estimate of the channel impulse responses as follows:

$$\hat{\boldsymbol{h}}(k+1) = \hat{\boldsymbol{h}}(k) - \mu\nabla J(k+1), \tag{51.26}$$

where μ is again a small positive step size. As shown in [51.30], the gradient of $J(k+1)$ is computed as:

$$\nabla J(k+1) = \frac{\partial J(k+1)}{\partial \hat{\boldsymbol{h}}(k)} = \frac{2\left[\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - J(k+1)\hat{\boldsymbol{h}}(k)\right]}{\|\hat{\boldsymbol{h}}(k)\|^2}\,, \tag{51.27}$$

where

$$\tilde{\boldsymbol{R}}_{x+}(k) \triangleq \begin{pmatrix} \sum_{n\neq 1} \tilde{\boldsymbol{R}}_{x_n x_n}(k) & -\tilde{\boldsymbol{R}}_{x_2 x_1}(k) & \cdots & -\tilde{\boldsymbol{R}}_{x_N x_1}(k) \\ -\tilde{\boldsymbol{R}}_{x_1 x_2}(k) & \sum_{n\neq 2} \tilde{\boldsymbol{R}}_{x_n x_n}(k) & \cdots & -\tilde{\boldsymbol{R}}_{x_N x_2}(k) \\ \vdots & \vdots & \ddots & \vdots \\ -\tilde{\boldsymbol{R}}_{x_1 x_N}(k) & -\tilde{\boldsymbol{R}}_{x_2 x_N}(k) & \cdots & \sum_{n\neq N} \tilde{\boldsymbol{R}}_{x_n x_n}(k) \end{pmatrix}$$

and

$$\tilde{\boldsymbol{R}}_{x_i x_j}(k) \triangleq \boldsymbol{x}_i(k)\boldsymbol{x}_j^{\mathrm{T}}(k)\,, \quad i, j = 1, 2, \cdots, N\,.$$

If the model filters are always normalized after each update, then a simplified MCLMS algorithm is obtained:

$$\hat{\boldsymbol{h}}(k+1) = \frac{\hat{\boldsymbol{h}}(k) - 2\mu\left[\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - J(k+1)\hat{\boldsymbol{h}}(k)\right]}{\left\|\hat{\boldsymbol{h}}(k) - 2\mu\left[\tilde{\boldsymbol{R}}_{x+}(k+1)\hat{\boldsymbol{h}}(k) - J(k+1)\hat{\boldsymbol{h}}(k)\right]\right\|}\,. \tag{51.28}$$

A number of other adaptive blind SIMO identification algorithms with faster convergence and lower computational complexity have also been developed, e.g., [51.31, 32]. Due to space limitations, we choose not to present the development of those algorithms and refer the reader to [51.33] and references therein for more details.

After the adaptive algorithm converges, the TDOA between any two microphones can be inferred as

$$\hat{\tau}_{ij}^{\mathrm{ABMCI}} = \arg\max_l |\hat{h}_{i,l}| - \arg\max_l |\hat{h}_{j,l}|\,, \quad i, j = 1, 2, \cdots, N\,, \tag{51.29}$$

which implies that in every channel the direct path is always dominant. This is generally true for acoustic waves, which would be considerably attenuated by wall reflection. But sometimes two or more reverberant signals via multipaths of equal delay could add coherently such that the direct-path component no longer dominates the impulse response. Therefore a more-robust way to pick the direct-path component is to identify the Q ($Q > 1$) strongest elements in the impulse responses and choose the one with the smallest delay [51.29]:

$$\hat{\tau}_{ij}^{\mathrm{ABMCI}} = \min\left\{\arg\max_l{}^q |\hat{h}_{i,l}|\right\} - \min\left\{\arg\max_l{}^q |\hat{h}_{j,l}|\right\}\,, \quad i, j = 1, 2, \cdots, N\,, \quad q = 1, 2, \cdots, Q\,, \tag{51.30}$$

where $\max^q$ computes the q-th largest element.

51.2.5 Multichannel Spatial Prediction and Interpolation Methods

While the blind SIMO identification-based TDE algorithms cope well with room reverberation, they have relatively poor tracking capability. As a result, in a rapidly varying acoustic environment, they will have a problem. From this perspective, cross-correlation-based methods do have some advantages and so there have been continuous efforts to improve their robustness against noise and reverberation.

In Sect. 51.2, only two microphones was considered. In this section, we explore the possibility of using multiple microphones (greater than two) to improve the estimate accuracy of the TDOA between the first two microphones in adverse acoustic environments. For a three-microphone system, the three TDOAs, namely τ_{12}, τ_{13}, and τ_{23}, are apparently not independent but obey the following relation: $\tau_{13} = \tau_{12} + \tau_{23}$. The idea of exploiting TDOA redundancy in sensor arrays was first conceived in [51.34] and a Kalman-filtering-based two-stage TDE algorithm was proposed. Recently, following a similar line of thoughts, several fusion algorithms have been developed [51.35–37]. In what follows, we present a multichannel TDE algorithm using spatial prediction and interpolation [51.38, 39], which takes advantage of the TDOA redundancy among multiple microphones in a more-intuitive way.

Suppose that there are N ($N > 2$) microphones in an open space to which the ideal free field can be properly applied. The N microphones are positioned in a prespecified geometric pattern such that the TDOA of a microphone pair can be expressed as a function of the TDOAs of a *small* number of other microphone pairs. The number depends on the distance of the source from the microphone array and on whether the microphones constitute a two-dimensional (2-D) or three-dimensional (3-D) array. For ease of presentation, let us consider a linear array and a far-field speech source. Then we

have

$$\tau_{1n} = \tau_1 - \tau_n = f_n(\tau_{12}) , \quad n = 1, 2, \cdots , N , \tag{51.31}$$

where

$$f_1(\tau_{12}) = 0 , \quad f_2(\tau_{12}) = \tau_{12} .$$

Substituting (51.31) into (51.1) gives

$$x_n(k) = \alpha_n s[k - \tau_1 + f_n(\tau_{12})] + b_n(k) . \tag{51.32}$$

In the absence of noise, since it can be easily checked that

$$x_n[k + f_N(\tau_{12}) - f_n(\tau_{12})] = \alpha_n s(k - \tau_N) , \quad \forall n = 1, 2, \cdots , N , \tag{51.33}$$

we know that $x_1[k + f_N(\tau_{12})]$ is aligned with $x_n[k + f_N(\tau_{12}) - f_n(\tau_{12})]$. Therefore, a forward spatial prediction error signal can be defined as:

$$e_1(k, \tau) \triangleq x_1[k + f_N(\tau)] - \boldsymbol{x}_{\mathrm{a},2:N}^{\mathrm{T}}(k, \tau)\boldsymbol{a}_{2:N}(\tau) , \tag{51.34}$$

where τ is a dummy variable for the hypothesized TDOA τ_{12},

$$\boldsymbol{x}_{\mathrm{a},2:N}(k, \tau) = \Big[x_2[k + f_N(\tau) - f_2(\tau)] \ \cdots \ x_N(k)\Big]^{\mathrm{T}} \tag{51.35}$$

is the aligned (subscript 'a') signal vector, and

$$\boldsymbol{a}_{2:N}(\tau) = \Big[a_2(\tau) \ a_3(\tau) \ \cdots \ a_N(\tau)\Big]^{\mathrm{T}}$$

contains the forward spatial linear prediction coefficients. Minimizing the mean-square value of the prediction error signal

$$J_1(\tau) \triangleq E\left\{\mathrm{e}_1^2(k, \tau)\right\} \tag{51.36}$$

leads to the Wiener–Hopf equation

$$\boldsymbol{R}_{\mathrm{a},2:N}(\tau)\boldsymbol{a}_{2:N}(\tau) = \boldsymbol{r}_{\mathrm{a},2:N}(\tau) , \tag{51.37}$$

where

$$\boldsymbol{R}_{\mathrm{a},2:N}(\tau) \triangleq E\left\{\boldsymbol{x}_{\mathrm{a},2:N}(k, \tau)\boldsymbol{x}_{\mathrm{a},2:N}^{\mathrm{T}}(k, \tau)\right\} = \begin{pmatrix} \sigma_{x_2}^2 & r_{\mathrm{a},x_2x_3}(\tau) & \cdots & r_{\mathrm{a},x_2x_N}(\tau) \\ r_{\mathrm{a},x_3x_2}(\tau) & \sigma_{x_3}^2 & \cdots & r_{\mathrm{a},x_3x_N}(\tau) \\ \vdots & \vdots & \ddots & \vdots \\ r_{\mathrm{a},x_Nx_2}(\tau) & r_{\mathrm{a},x_Nx_3}(\tau) & \cdots & \sigma_{x_N}^2 \end{pmatrix}$$

is the spatial correlation matrix of the aligned signals with

$$\sigma_{x_n}^2 = E\left\{x_n^2(k)\right\} , \quad n = 1, 2, \cdots , N ,$$

$$\begin{aligned} r_{\mathrm{a},x_ix_j}(\tau) &= E\big\{x_i[k + f_N(\tau) - f_i(\tau)] \\ &\quad \times x_j[k + f_N(\tau) - f_j(\tau)]\big\} \\ &= E\big\{x_i[k - f_i(\tau)]x_j[k - f_j(\tau)]\big\} , \\ i, j &= 1, 2, \cdots , N , \end{aligned}$$

and

$$\boldsymbol{r}_{\mathrm{a},2:N}(\tau) \triangleq \Big[r_{\mathrm{a},x_1x_2}(\tau) \ r_{\mathrm{a},x_1x_3}(\tau) \ \cdots \ r_{\mathrm{a},x_1x_N}(\tau)\Big]^{\mathrm{T}} .$$

Substituting the solution of the Wiener–Hopf (51.37), which is

$$\boldsymbol{a}_{2:N}(\tau) = \boldsymbol{R}_{\mathrm{a},2:N}^{-1}(\tau)\boldsymbol{r}_{\mathrm{a},2:N}(\tau) ,$$

into (51.34) gives the minimum forward prediction error

$$\begin{aligned} &e_{1,\min}(k, \tau) \\ &= x_1[k + f_N(\tau)] - \boldsymbol{x}_{\mathrm{a},2:N}^{\mathrm{T}}(k, \tau)\boldsymbol{R}_{\mathrm{a},2:N}^{-1}(\tau)\boldsymbol{r}_{\mathrm{a},2:N}(\tau) . \end{aligned} \tag{51.38}$$

Accordingly, we have

$$\begin{aligned} J_{1,\min}(\tau) &\triangleq E\left\{e_{1,\min}^2(k, \tau)\right\} \\ &= \sigma_{x_1}^2 - \boldsymbol{r}_{\mathrm{a},2:N}^{\mathrm{T}}(\tau)\boldsymbol{R}_{\mathrm{a},2:N}^{-1}(\tau)\boldsymbol{r}_{\mathrm{a},2:N}(\tau) . \end{aligned} \tag{51.39}$$

Then we can argue that the time lag τ that induces a minimum $J_{1,\min}(\tau)$ would be the TDOA between the first two microphones:

$$\hat{\tau}_{12}^{\mathrm{FSP}} = \arg\min_{\tau} J_{1,\min}(\tau) , \tag{51.40}$$

where the superscript 'FSP' stands for forward spatial predication .

Similarly, the multichannel TDE algorithm can be developed using backward prediction or interpolation with any one of the N microphone outputs being regarded as the reference signal.

51.2.6 Multichannel Cross-Correlation Coefficient Algorithm

The idea of using the spatial prediction (or interpolation) error to measure the correlation among multiple signals is intuitive and helpful. However in statistics and signal processing, the cross-correlation coefficient is a more widely used concept for this purpose. Since the cross-correlation coefficient is defined traditionally only for

two random processes, it first needs to be generalized to multiple random processes before being employed in the development of a multichannel TDE algorithm.

The definition of a multiple coherence function, derived from the concepts of the ordinary coherence function between two signals and the partial (conditioned) coherence function, was presented in [51.40] to measure the correlation among the output of a multiple-input signal-output (MISO) system and its inputs. In [51.38] and [51.39], the multichannel cross-correlation coefficient (MCCC) was constructed in a new way from the multichannel correlation matrix; this provides a seamless generalization of the classical cross-correlation coefficient to the case where there are multiple random processes.

Following (51.35), we denote

$$\boldsymbol{x}_{\mathrm{a}}(k,\tau) \triangleq \left[x_1[k+f_N(\tau)] \;\; \boldsymbol{x}_{\mathrm{a},2:N}^{\mathrm{T}}(k,\tau)\right]^{\mathrm{T}} , \quad (51.41)$$

and

$$\boldsymbol{R}_{\mathrm{a}}(\tau) \triangleq E\left\{\boldsymbol{x}_{\mathrm{a}}(k,\tau)\boldsymbol{x}_{\mathrm{a}}^{\mathrm{T}}(k,\tau)\right\} = \begin{pmatrix} \sigma_{x_1}^2 & r_{\mathrm{a},x_1x_2}(\tau) & \cdots & r_{\mathrm{a},x_1x_N}(\tau) \\ r_{\mathrm{a},x_2x_1}(\tau) & \sigma_{x_2}^2 & \cdots & r_{\mathrm{a},x_2x_N}(\tau) \\ \vdots & \vdots & \ddots & \vdots \\ r_{\mathrm{a},x_Nx_1}(\tau) & r_{\mathrm{a},x_Nx_2}(\tau) & \cdots & \sigma_{x_N}^2 \end{pmatrix} . \quad (51.42)$$

The spatial correlation matrix $\boldsymbol{R}_{\mathrm{a}}(\tau)$ can be factored as follows:

$$\boldsymbol{R}_{\mathrm{a}}(\tau) = \boldsymbol{\Sigma}\tilde{\boldsymbol{R}}_{\mathrm{a}}(\tau)\boldsymbol{\Sigma}, \quad (51.43)$$

where

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{x_1} & 0 & \cdots & 0 \\ 0 & \sigma_{x_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \sigma_{x_N} \end{pmatrix}$$

is a diagonal matrix,

$$\tilde{\boldsymbol{R}}_{\mathrm{a}}(\tau) = \begin{pmatrix} 1 & \rho_{\mathrm{a},x_1x_2}(\tau) & \cdots & \rho_{\mathrm{a},x_1x_N}(\tau) \\ \rho_{\mathrm{a},x_2x_1}(\tau) & 1 & \cdots & \rho_{\mathrm{a},x_2x_N}(\tau) \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{\mathrm{a},x_Nx_1}(\tau) & \rho_{\mathrm{a},x_Nx_2}(\tau) & \cdots & 1 \end{pmatrix}$$

is a symmetric matrix, and

$$\rho_{\mathrm{a},x_ix_j}(\tau) \triangleq \frac{r_{\mathrm{a},x_ix_j}(\tau)}{\sigma_i\sigma_j}, \quad i,j = 1,2,\cdots,N ,$$

is the cross-correlation coefficient between the i-th and j-th aligned microphone signals.

Since matrix $\tilde{\boldsymbol{R}}_{\mathrm{a}}(\tau)$ is symmetric, positively semidefinite, and its diagonal elements are all equal to one, it was shown in [51.38] and [51.39] that

$$0 \le \det\left[\tilde{\boldsymbol{R}}_{\mathrm{a}}(\tau)\right] \le 1 , \quad (51.44)$$

where $\det(\cdot)$ stands for determinant.

In the two-channel case, it can easily be checked that the squared cross-correlation coefficient is linked to the normalized spatial correlation matrix by

$$\rho_{\mathrm{a},x_1x_2}^2(\tau) = 1 - \det\left[\tilde{\boldsymbol{R}}_{\mathrm{a},1:2}(\tau)\right] . \quad (51.45)$$

Then by analogy the squared MCCC among the N aligned signals $x_n[k+f_N(\tau)-f_n(\tau)]$, $n=1,2,\cdots,N$, is constructed as:

$$\rho_{\mathrm{a},x_1:x_N}^2(\tau) \triangleq 1 - \det\left[\tilde{\boldsymbol{R}}_{\mathrm{a}}(\tau)\right] . = 1 - \frac{\det\left[\boldsymbol{R}_{\mathrm{a}}(\tau)\right]}{\prod_{n=1}^{N}\sigma_{x_n}^2} . \quad (51.46)$$

The MCCC has the following properties (presented without proof) [51.41].

1. $0 \le \rho_{\mathrm{a},x_1:x_N}^2(\tau) \le 1$.
2. If two or more signals are perfectly correlated, then $\rho_{\mathrm{a},x_1:x_N}^2(\tau) = 1$.
3. If all the signals are completely uncorrelated with each other, then $\rho_{\mathrm{a},x_1:x_N}^2(\tau) = 0$.
4. If one of the signals is completely uncorrelated with the $N-1$ other signals, then the MCCC will measure the correlation among those $N-1$ remaining signals.

Using the definition of MCCC, we deduce an estimate of the TDOA between the first two microphone signals as

$$\hat{\tau}_{12}^{\mathrm{MCCC}} = \arg\max_{\tau} \rho_{\mathrm{a},x_1:x_N}^2(\tau) , \quad (51.47)$$

which is equivalent to computing

$$\hat{\tau}_{12}^{\mathrm{MCCC}} = \arg\max_{\tau}\left\{1 - \frac{\det\left[\boldsymbol{R}_{\mathrm{a}}(\tau)\right]}{\prod_{n=1}^{N}\sigma_{x_n}^2}\right\} = \arg\min_{\tau}\det\left[\boldsymbol{R}_{\mathrm{a}}(\tau)\right] . \quad (51.48)$$

To investigate the link of the MCCC method to the FSP method, let us revisit the spatial prediction error function given by (51.39). We define

$$\boldsymbol{a}(\tau) \triangleq \left[a_1(\tau) \;\; a_2(\tau) \;\; \cdots \;\; a_N(\tau)\right]^{\mathrm{T}} = \left[a_1(\tau) \;\; \boldsymbol{a}_{2:N}^{\mathrm{T}}(\tau)\right]^{\mathrm{T}} . \quad (51.49)$$

Then, for $a_1(\tau) = -1$, the forward spatial prediction error signal (51.34) can be written as

$$e_1(k,\tau) = -\boldsymbol{x}_{\mathrm{a}}^{\mathrm{T}}(k,\tau)\boldsymbol{a}(\tau)\,, \tag{51.50}$$

and (51.36) can be expressed as

$$\begin{aligned} J_1(\tau) &= E\left\{\mathrm{e}_1^2(k,\tau)\right\} + \kappa\left[\boldsymbol{v}_1^{\mathrm{T}}\boldsymbol{a}(\tau) + 1\right] \\ &= \boldsymbol{a}^{\mathrm{T}}(\tau)\boldsymbol{R}_{\mathrm{a}}(\tau)\boldsymbol{a}(\tau) + \kappa\left[\boldsymbol{v}_1^{\mathrm{T}}\boldsymbol{a}(\tau) + 1\right]\,, \end{aligned} \tag{51.51}$$

where κ is a Lagrange multiplier introduced to force $a_1(\tau)$ to have value -1 and

$$\boldsymbol{v}_1 \triangleq \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^{\mathrm{T}}\,.$$

Taking the derivative of (51.51) with respect to $\boldsymbol{a}(\tau)$ and setting the result to zero yields

$$\frac{\partial J_1(\tau)}{\partial \boldsymbol{a}(\tau)} = 2\boldsymbol{R}_{\mathrm{a}}(\tau)\boldsymbol{a}(\tau) + k\boldsymbol{v}_1 = \boldsymbol{0}\,. \tag{51.52}$$

Solving (51.52) for $\boldsymbol{a}(\tau)$ produces

$$\boldsymbol{a}(\tau) = -\frac{\kappa\boldsymbol{R}_{\mathrm{a}}^{-1}(\tau)\boldsymbol{v}_1}{2}\,. \tag{51.53}$$

Substituting (51.53) into (51.51) leads to

$$J_1(\tau) = \kappa\left(1 - \frac{\boldsymbol{v}_1^{\mathrm{T}}\boldsymbol{R}_{\mathrm{a}}^{-1}(\tau)\boldsymbol{v}_1}{4}\kappa\right)\,, \tag{51.54}$$

from which we know that

$$J_{1,\min}(\tau) = \frac{1}{\boldsymbol{v}_1^{\mathrm{T}}\boldsymbol{R}_{\mathrm{a}}^{-1}(\tau)\boldsymbol{v}_1}\,. \tag{51.55}$$

Substituting (51.43) into (51.55) and using the fact that

$$\boldsymbol{\Sigma}^{-1}\boldsymbol{v}_1 = \frac{\boldsymbol{v}_1}{\sigma_{x_1}}\,,$$

we have

$$J_{1,\min}(\tau) = \frac{\sigma_{x_1}^2}{\boldsymbol{v}_1^{\mathrm{T}}\tilde{\boldsymbol{R}}_{\mathrm{a}}^{-1}(\tau)\boldsymbol{v}_1}\,. \tag{51.56}$$

Note that $\boldsymbol{v}_1^{\mathrm{T}}\tilde{\boldsymbol{R}}_{\mathrm{a}}^{-1}(\tau)\boldsymbol{v}_1$ is the (1, 1)-th element of matrix $\tilde{\boldsymbol{R}}_{\mathrm{a}}^{-1}(\tau)$, which is computed using the adjoint method as the (1, 1)-th cofactor of $\tilde{\boldsymbol{R}}_{\mathrm{a}}(\tau)$ divided by the determinant of $\tilde{\boldsymbol{R}}_{\mathrm{a}}(\tau)$, i. e.,

$$\boldsymbol{v}_1^{\mathrm{T}}\tilde{\boldsymbol{R}}_{\mathrm{a}}^{-1}(\tau)\boldsymbol{v}_1 = \frac{\det\left[\tilde{\boldsymbol{R}}_{\mathrm{a},2:\mathrm{N}}(\tau)\right]}{\det\left[\tilde{\boldsymbol{R}}_{\mathrm{a}}(\tau)\right]}\,, \tag{51.57}$$

where $\tilde{\boldsymbol{R}}_{\mathrm{a},2:\mathrm{N}}(\tau)$ is the lower-right submatrix of $\tilde{\boldsymbol{R}}_{\mathrm{a}}(\tau)$ formed by removing the first row and the first column. By substituting (51.57) into (51.56), we get

$$J_{1,\min}(\tau) = \sigma_{x_1}^2 \cdot \frac{\det\left[\tilde{\boldsymbol{R}}_{\mathrm{a}}(\tau)\right]}{\det\left[\tilde{\boldsymbol{R}}_{\mathrm{a},2:\mathrm{N}}(\tau)\right]}\,. \tag{51.58}$$

Therefore, the FSP estimate of τ_{12} is found as

$$\begin{aligned} \hat{\tau}_{12}^{\mathrm{FSP}} &= \arg\min_{\tau} J_{1,\min}(\tau) \\ &= \arg\min_{\tau} \frac{\det\left[\tilde{\boldsymbol{R}}_{\mathrm{a}}(\tau)\right]}{\det\left[\tilde{\boldsymbol{R}}_{\mathrm{a},2:\mathrm{N}}(\tau)\right]}\,. \end{aligned} \tag{51.59}$$

Comparing (51.48) to (51.59) reveals a clear distinction between the two methods in spite of the strong similarity in principle. In practice, the FSP method has a numerical stability problem since the calculation of the FSP cost function (51.58) involves division by $\det\left[\tilde{\boldsymbol{R}}_{\mathrm{a},2:N}(\tau)\right]$, while the MCCC method is found to be fairly stable.

It is worth pointing out that the microphone outputs can be pre-whitened before computing their MCCC as was done in the SCOT algorithm in the two-channel scenario. Using this approach, the TDE algorithms become more robust to volume variations of the speech source signal.

51.2.7 Minimum-Entropy Method

While the MCCC-based TDE algorithm performs well in the presence of noise and reverberation, the MCCC is by no means the only choice for developing the concept of multichannel TDE. MCCC is a second-order-statistics (SOS) measure of dependence among multiple random variables and is ideal for Gaussian source signals. However for non-Gaussian source signals such as speech, MCCC is not sufficient and higher-order statistics (HOS) have more to say about their dependence.

The concept of entropy, which is a statistical (apparently HOS) measure of randomness or uncertainty of a random variable, was introduced by *Shannon* in the context of communication theory [51.42]. For a random variable x with a probability density function (PDF) $p(x)$ (in this chapter, we choose not to distinguish between random variables and their realizations for the conciseness of presentation), the entropy is defined as [51.43]:

$$\begin{aligned} H(x) &= -\int p(x)\ln p(x)\,\mathrm{d}x \\ &= -E\left\{\ln p(x)\right\}\,. \end{aligned} \tag{51.60}$$

The entropy (in the continuous case) is a measure of the structure contained in the PDF [51.44]. As far as the

multivariate random variable $\boldsymbol{x}_{\mathrm{a}}(k,\tau)$ given by (51.41) is concerned, the joint entropy is:

$$H\left[\boldsymbol{x}_{\mathrm{a}}(k,\tau)\right] = -\int p\left[\boldsymbol{x}_{\mathrm{a}}(k,\tau)\right] \ln p\left[\boldsymbol{x}_{\mathrm{a}}(k,\tau)\right] \mathrm{d}\boldsymbol{x}_{\mathrm{a}}(k,\tau)\,. \tag{51.61}$$

It was then argued in [51.45] that the time lag τ that gives the minimum of $H\left[\boldsymbol{x}_{\mathrm{a}}(k,\tau)\right]$ corresponds to the TDOA between the two microphones:

$$\hat{\tau}_{12}^{\mathrm{ME}} = \arg\min_{\tau} H\left[\boldsymbol{x}_{\mathrm{a}}(k,\tau)\right], \tag{51.62}$$

where the superscript 'ME' refers to the minimum-entropy method.

Gaussian Source Signal

If the source is Gaussian, so are the microphone outputs in the absence of noise. Suppose that the aligned microphone signals are zero-mean and joint Gaussian random signals. Their joint PDF is then given by

$$p\left[\boldsymbol{x}_{\mathrm{a}}(k,\tau)\right] = \frac{\exp\left[-\eta_{\mathrm{a}}(k,\tau)/2\right]}{\sqrt{(2\pi)^N \det\left[\boldsymbol{R}_{\mathrm{a}}(\tau)\right]}}\,, \tag{51.63}$$

where

$$\eta_{\mathrm{a}}(k,\tau) \triangleq \boldsymbol{x}_{\mathrm{a}}^{\mathrm{T}}(k,\tau)\boldsymbol{R}_{\mathrm{a}}^{-1}(\tau)\boldsymbol{x}_{\mathrm{a}}(k,\tau)\,. \tag{51.64}$$

By substituting (51.63) into (51.61), the joint entropy can be computed [51.45] as:

$$H\left[\boldsymbol{x}_{\mathrm{a}}(k,\tau)\right] = \frac{1}{2}\ln\left\{(2\pi\,\mathrm{e})^N \det\left[\boldsymbol{R}_{\mathrm{a}}(\tau)\right]\right\}\,. \tag{51.65}$$

Consequently, (51.62) becomes

$$\hat{\tau}_{12}^{\mathrm{ME}} = \arg\min_{\tau} \det\left[\boldsymbol{R}_{\mathrm{a}}(\tau)\right]\,. \tag{51.66}$$

It is clear from (51.48) and (51.66) that minimizing the entropy is equivalent to maximizing the MCCC for Gaussian source signals.

Speech Source Signal

Speech is a complicated random process and there is no rigorous mathematical formula for its entropy. However, in speech research it was found that speech can be fairly well modeled by a Laplace distribution [51.46, 47].

The univariate Laplace distribution with zero mean and variance σ_x^2 is given by

$$p(x) = \frac{\sqrt{2}}{2\sigma_x}\,\mathrm{e}^{-\sqrt{2}|x|/\sigma_x}\,, \tag{51.67}$$

and the corresponding entropy is [51.43],

$$H(x) = 1 + \ln\left(\sqrt{2}\,\sigma_x\right). \tag{51.68}$$

Suppose that $\boldsymbol{x}_{\mathrm{a}}(k,\tau)$ has a multivariate Laplace distribution with mean $\boldsymbol{0}$ and covariance matrix $\boldsymbol{R}_{\mathrm{a}}(\tau)$ [51.48], [51.49]:

$$p\left[\boldsymbol{x}_{\mathrm{a}}(k,\tau)\right] = \frac{2\left[\eta_{\mathrm{a}}(k,\tau)/2\right]^{P/2} K_P\left[\sqrt{2\eta_{\mathrm{a}}(k,\tau)}\right]}{\sqrt{(2\pi)^N \det\left[\boldsymbol{R}_{\mathrm{a}}(\tau)\right]}}\,, \tag{51.69}$$

where $P = (2-N)/2$ and $K_P(\cdot)$ is the modified Bessel function of the third kind (also called the modified Bessel function of the second kind) given by

$$K_P(a) = \frac{1}{2}\left(\frac{a}{2}\right)^P \int_0^\infty z^{-P-1}\exp\left(-z - \frac{a^2}{4z}\right)\mathrm{d}z\,, \quad a > 0\,. \tag{51.70}$$

The joint entropy is

$$H\left[\boldsymbol{x}_{\mathrm{a}}(k,\tau)\right] = \frac{1}{2}\ln\left(\frac{(2\pi)^N \det\left[\boldsymbol{R}_{\mathrm{a}}(\tau)\right]}{4}\right) - \frac{P}{2}E\left\{\ln\left(\frac{\eta_{\mathrm{a}}(k,\tau)}{2}\right)\right\} - E\left\{\ln K_P\left[\sqrt{2\eta_{\mathrm{a}}(k,\tau)}\right]\right\}\,. \tag{51.71}$$

The two quantities $E\left\{\ln\left[\eta_{\mathrm{a}}(k,\tau)/2\right]\right\}$ and $E\left\{\ln K_P\left[\sqrt{2\eta_{\mathrm{a}}(k,\tau)}\right]\right\}$ do not seem to have a closed form. So a numerical scheme needs to be developed to estimate them. One possibility to do this is the following. Assume that all processes are ergodic . As a result, ensemble averages can be replaced by time averages. If there are K samples for each element of the observation vector $\boldsymbol{x}_{\mathrm{a}}(k,\tau)$, the following estimators were proposed in [51.45]:

$$E\left\{\ln\left[\eta_{\mathrm{a}}(k,\tau)/2\right]\right\} \approx \frac{1}{K}\sum_{k=0}^{K-1}\ln\left[\eta_{\mathrm{a}}(k,\tau)/2\right], \tag{51.72}$$

$$E\left\{\ln K_P\left[\sqrt{2\eta_{\mathrm{a}}(k,\tau)}\right]\right\} \approx \frac{1}{K}\sum_{k=0}^{K-1}\ln K_P\left[\sqrt{2\eta_{\mathrm{a}}(k,\tau)}\right]\,. \tag{51.73}$$

The simulation results presented in [51.45] show that the ME algorithm performs in general comparably to or better than the MCCC algorithm. Apparently the ME algorithm is computationally intensive, but the idea of using entropy expands our horizon of knowledge in pursuit of new TDE algorithms.

51.3 Source Localization

51.3.1 Problem Formulation

The problem addressed here is the estimation of the location of an acoustic point source given the array geometry and the relative TDOA measurements between different microphone pairs. The problem is stated mathematically as follows.

Consider an array that consists of N microphones. As shown in Fig. 51.2, these microphones are located in a 3-D Cartesian coordinate system at positions

$$\boldsymbol{\gamma}_n \triangleq \begin{bmatrix} x_n & y_n & z_n \end{bmatrix}^{\mathrm{T}}, \quad n = 1, 2, \cdots, N. \tag{51.74}$$

Without loss of generality, the first microphone ($n = 1$) is regarded as the reference and is placed at the origin of the coordinate system, i. e., $\boldsymbol{\gamma}_1 = [0\ 0\ 0]^{\mathrm{T}}$. The acoustic source is located at $\boldsymbol{\gamma}_{\mathrm{s}} \triangleq [x_{\mathrm{s}}\ y_{\mathrm{s}}\ z_{\mathrm{s}}]^{\mathrm{T}}$. The distances from the origin to the n-th microphone and the source are denoted by Ω_n and Ω_{s}, respectively, where

$$\Omega_n \triangleq \|\boldsymbol{\gamma}_n\| = \sqrt{x_n^2 + y_n^2 + z_n^2}, \quad n = 1, 2, \cdots, N, \tag{51.75}$$

$$\Omega_{\mathrm{s}} \triangleq \|\boldsymbol{\gamma}_{\mathrm{s}}\| = \sqrt{x_{\mathrm{s}}^2 + y_{\mathrm{s}}^2 + z_{\mathrm{s}}^2}. \tag{51.76}$$

The distance between the source and the n-th microphone is denoted by

$$D_n \triangleq \|\boldsymbol{\gamma}_n - \boldsymbol{\gamma}_{\mathrm{s}}\| = \sqrt{(x_n - x_{\mathrm{s}})^2 + (y_n - y_{\mathrm{s}})^2 + (z_n - z_{\mathrm{s}})^2}. \tag{51.77}$$

The difference in the distances of microphones i and j from the source is given by

$$d_{ij} \triangleq D_i - D_j, \quad i, j = 1, 2, \cdots, N. \tag{51.78}$$

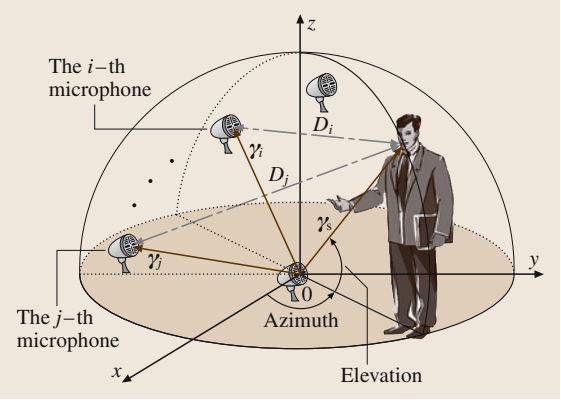


Fig. 51.2 Spatial diagram illustrating variables defined in the source localization problem

This difference is usually termed the range difference; it is proportional to the time difference of arrival τ_{ij}. If the speed of sound is c, then

$$d_{ij} = c \cdot \tau_{ij}. \tag{51.79}$$

The speed of sound (in m/s) can be estimated from the air temperature t_{air} (in degrees Celsius) according to the following approximate (first-order) formula,

$$c \approx 331 + 0.610\, t_{\mathrm{air}}. \tag{51.80}$$

The localization problem is then to estimate $\boldsymbol{\gamma}_{\mathrm{s}}$ given the set of $\boldsymbol{\gamma}_n$ and τ_{ij}. Note that there are $N(N-1)/2$ distinct TDOA estimates τ_{ij}, which exclude the case $i = j$ and counts the $\tau_{ij} = -\tau_{ji}$ pair only once. However, in the absence of noise, the space spanned by these TDOA estimates is $(N-1)$-dimensional. Any $N-1$ linearly independent TDOAs determine all of the others. In a noisy environment, the TDOA redundancy can be used to improve the accuracy of the source localization algorithms, but this would increase their computational complexity. For simplicity and also without loss of generality, we choose $\tau_{n1}, n = 2, \cdots, N$ as the basis for this $\mathbb{R}^{N-1}$ space in this Chapter.

51.3.2 Measurement Model and Cramèr–Rao Lower Bound

When the source localization problem is examined using estimation theory, the measurements of the range differences are modeled by

$$d_{n1} = g_n(\boldsymbol{\gamma}_{\mathrm{s}}) + \epsilon_n, \quad n = 2, \cdots, N, \tag{51.81}$$

where

$$g_n(\boldsymbol{\gamma}_{\mathrm{s}}) = \|\boldsymbol{\gamma}_n - \boldsymbol{\gamma}_{\mathrm{s}}\| - \|\boldsymbol{\gamma}_{\mathrm{s}}\|,$$

and the ϵ_n are measurement errors. In vector form, such an additive measurement error model becomes,

$$\boldsymbol{d} = \boldsymbol{g}(\boldsymbol{\gamma}_{\mathrm{s}}) + \boldsymbol{\epsilon}, \tag{51.82}$$

where

$$\boldsymbol{d} = \begin{bmatrix} d_{21} & d_{31} & \cdots & d_{N1} \end{bmatrix}^{\mathrm{T}},$$

$$\boldsymbol{g}(\boldsymbol{\gamma}_{\mathrm{s}}) = \begin{bmatrix} g_2(\boldsymbol{\gamma}_{\mathrm{s}}) & g_3(\boldsymbol{\gamma}_{\mathrm{s}}) & \cdots & g_N(\boldsymbol{\gamma}_{\mathrm{s}}) \end{bmatrix}^{\mathrm{T}},$$

$$\boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_2 & \epsilon_3 & \cdots & \epsilon_N \end{bmatrix}^{\mathrm{T}}.$$

Furthermore, it was postulated that the additive measurement errors have zero mean and are independent of

the range difference observation, as well as the source location $\boldsymbol{\gamma}_s$ [51.50]. For a continuous-time estimator, the corrupting noise, as indicated in [51.16], is jointly Gaussian distributed. The PDF of $\boldsymbol{d}$ conditioned on $\boldsymbol{\gamma}_s$ is subsequently given by,

$$p\left(\boldsymbol{d}|\boldsymbol{\gamma}_s\right) = \frac{\exp\left\{-\frac{1}{2}\left[\boldsymbol{d}-\boldsymbol{g}(\boldsymbol{\gamma}_s)\right]^T \boldsymbol{R}_{\epsilon\epsilon}^{-1}\left[\boldsymbol{d}-\boldsymbol{g}(\boldsymbol{\gamma}_s)\right]\right\}}{\sqrt{(2\pi)^N \det(\boldsymbol{R}_{\epsilon\epsilon})}}, \tag{51.83}$$

where $\boldsymbol{R}_{\epsilon\epsilon} \triangleq E\left\{\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T\right\}$ is the covariance matrix of $\boldsymbol{\epsilon}$.

Note that $\boldsymbol{R}_{\epsilon\epsilon}$ is independent of $\boldsymbol{\gamma}_s$ by assumption. Since digital equipment is used to sample the microphone waveforms and estimate the TDOAs, the error introduced by discrete-time processing also has to be taken into account. When this is done, the measurement error is no longer Gaussian and is more-properly modeled as a mixture of Gaussian noise and noise that is uniformly distributed over $[-T_s c/2, T_s c/2]$, where T_s is the sampling period. As an example, for a digital source location estimator with an 8 KHz sampling rate operating at room temperature (25 °C, implying that $c \approx 346.25$ m/s), the maximum error in range difference estimates due to sampling is about ± 2.164 cm, which leads to considerable errors in the location estimate, especially when the source is far from the microphone array.

Under the measurement model given by (51.82), we are now faced with the parameter estimation problem of extracting the source location information from the mismeasured range differences or the equivalent TDOAs. For an *unbiased* estimator, a Cramèr–Rao lower bound (CRLB) can be placed on the variance of each estimated coordinate of the source location. However, since the range difference function $\boldsymbol{g}(\boldsymbol{\gamma}_s)$ in the measurement model is nonlinear in the parameters under estimation, it is very difficult (or even impossible) to find an unbiased estimator that is mathematically simple and attains the CRLB. The CRLB is usually used as a benchmark against which the statistical efficiency of any unbiased estimators can be compared.

In general, without any assumptions made about the PDF of the measurement error $\boldsymbol{\epsilon}$, the CRLB of the i-th ($i = 1, 2, 3$) parameter variance is found as the $[i, i]$ element of the inverse of the Fisher information matrix defined by [51.51]:

$$[\boldsymbol{I}(\boldsymbol{\gamma}_s)]_{ij} \triangleq -E\left(\frac{\partial^2 \ln p\left(\boldsymbol{d}|\boldsymbol{\gamma}_s\right)}{\partial\gamma_{s,i}\partial\gamma_{s,j}}\right), \quad i, j = 1, 2, 3, \tag{51.84}$$

where the three parameters of $\boldsymbol{\gamma}_s$, i. e., $\gamma_{s,1}$, $\gamma_{s,2}$, and $\gamma_{s,3}$, are the x, y, and z coordinates of the source location, respectively.

In the case of a Gaussian measurement error, the Fisher information matrix becomes [51.52]:

$$\boldsymbol{I}(\boldsymbol{\gamma}_s) = \left(\frac{\partial \boldsymbol{g}(\boldsymbol{\gamma}_s)}{\partial \boldsymbol{\gamma}_s}\right)^T \boldsymbol{R}_{\epsilon\epsilon}^{-1}\left(\frac{\partial \boldsymbol{g}(\boldsymbol{\gamma}_s)}{\partial \boldsymbol{\gamma}_s}\right), \tag{51.85}$$

where $\partial \boldsymbol{g}(\boldsymbol{\gamma}_s)/\partial \boldsymbol{\gamma}_s$ is an $(N-1)\times 3$ Jacobian matrix defined as,

$$\frac{\partial \boldsymbol{g}(\boldsymbol{\gamma}_s)}{\partial \boldsymbol{\gamma}_s} = \begin{pmatrix} \frac{\partial g_2(\boldsymbol{\gamma}_s)}{\partial x_s} & \frac{\partial g_2(\boldsymbol{\gamma}_s)}{\partial y_s} & \frac{\partial g_2(\boldsymbol{\gamma}_s)}{\partial z_s} \\ \frac{\partial g_3(\boldsymbol{\gamma}_s)}{\partial x_s} & \frac{\partial g_3(\boldsymbol{\gamma}_s)}{\partial y_s} & \frac{\partial g_3(\boldsymbol{\gamma}_s)}{\partial z_s} \\ \vdots & \vdots & \vdots \\ \frac{\partial g_N(\boldsymbol{\gamma}_s)}{\partial x_s} & \frac{\partial g_N(\boldsymbol{\gamma}_s)}{\partial y_s} & \frac{\partial g_N(\boldsymbol{\gamma}_s)}{\partial z_s} \end{pmatrix} = \begin{pmatrix} (\boldsymbol{u}_{2\to s} - \boldsymbol{u}_{1\to s})^T \\ (\boldsymbol{u}_{3\to s} - \boldsymbol{u}_{1\to s})^T \\ \vdots \\ (\boldsymbol{u}_{N\to s} - \boldsymbol{u}_{1\to s})^T \end{pmatrix}, \tag{51.86}$$

and

$$\boldsymbol{u}_{n\to s} \triangleq \frac{\boldsymbol{\gamma}_s - \boldsymbol{\gamma}_n}{\|\boldsymbol{\gamma}_s - \boldsymbol{\gamma}_n\|} = \frac{\boldsymbol{\gamma}_s - \boldsymbol{\gamma}_n}{D_n}, \quad n = 1, 2, \cdots, N, \tag{51.87}$$

is the normalized vector of unit length pointing from the n-th microphone to the sound source.

51.3.3 Maximum-Likelihood Estimator

The measurement model for the source localization problem is apparently nonlinear; an efficient estimator that attains the CRLB may not exist or otherwise might be impossible to identify. In practice, the maximum-likelihood estimator is often used since it has the well-proven advantage of asymptotic efficiency for a large sample space.

To apply the maximum-likelihood principle, the statistical characteristics of the measurements need to be known or properly assumed prior to any processing. From the central limit theorem and also for mathematical simplicity, the measurement error is usually modeled as Gaussian and the likelihood function is given by (51.83), which is considered as a function of the source position $\boldsymbol{\gamma}_s$ under estimation.

Since the exponential function is monotonically increasing, the MLE is equivalent to minimizing a (log-

likelihood) cost function defined as,

$$\mathcal{L}(\boldsymbol{\gamma}_s) \triangleq \left[\boldsymbol{d} - \boldsymbol{g}(\boldsymbol{\gamma}_s)\right]^T \boldsymbol{R}_{\epsilon\epsilon}^{-1} \left[\boldsymbol{d} - \boldsymbol{g}(\boldsymbol{\gamma}_s)\right] . \quad (51.88)$$

Direct estimation of the minimizer is generally not practical. If the noise signals at different microphones are assumed to be uncorrelated, the covariance matrix is diagonal

$$\boldsymbol{R}_{\epsilon\epsilon} = \mathrm{diag}(\sigma_{\epsilon_2}^2, \sigma_{\epsilon_3}^2, \cdots, \sigma_{\epsilon_N}^2), \quad (51.89)$$

where $\sigma_{\epsilon_n}^2 = E\{\epsilon_n^2\}$ $(n = 2, 3, \cdots, N)$ is the variance of ϵ_n, and the cost function (51.88) becomes,

$$\mathcal{L}(\boldsymbol{\gamma}_s) = \sum_{n=2}^{N} \frac{\left[d_{n1} - g_n(\boldsymbol{\gamma}_s)\right]^2}{\sigma_{\epsilon_n}^2}. \quad (51.90)$$

Among other approaches, the steepest-descent algorithm can be used to find $\hat{\boldsymbol{\gamma}}_s^{\mathrm{MLE}}$ iteratively with

$$\hat{\boldsymbol{\gamma}}_s(k+1) = \hat{\boldsymbol{\gamma}}_s(k) - \frac{1}{2}\mu \nabla \mathcal{L}[\hat{\boldsymbol{\gamma}}_s(k)] , \quad (51.91)$$

where μ is the step size.

The foregoing MLE can be determined and is asymptotically optimal for this problem only if its two assumptions (Gaussian and uncorrelated measurement noise) hold. However, this is not the case in practice, as discussed in Sect. 51.3. Furthermore, the number of microphones in an array for camera pointing or beamformer steering is always limited, which makes the source localization a finite-sample rather than a large-sample problem. In addition, the cost function (51.90) is generally not strictly convex. In order to avoid a local minimum with the steepest-descent algorithm, we need to select a good initial guesstimate of the source location, which is difficult to do in practice, and convergence of the iterative algorithm to the desired solution cannot be guaranteed.

51.3.4 Least-Squares Estimators

Two limitations of the MLE are that probabilistic assumptions have to be made about the measured range differences and that the iterative algorithm to find the solution is computationally intensive. An alternative method is the well-known least-squares estimator (LSE). The LSE makes no probabilistic assumptions about the data and hence can be applied to the source localization problem in which a precise statistical characterization of the data is hard to determine. Furthermore, an LSE usually produces a closed-form estimate, which is desirable in real-time applications.

51.3.5 Least-Squares Error Criteria

In the least-squares (LS) approach, we attempt to minimize a squared error function that is zero in the absence of noise and model inaccuracies. Various error functions can be defined for closeness from the assumed (noiseless) signal based on hypothesized parameters for the observed data. When these are applied, various LSEs can be derived. For the source localization problem two LS error criteria have been proposed.

Hyperbolic LS Error Function

The first LS error function is defined as the difference between the observed range difference and that generated by a signal model depending upon the unknown parameters:

$$\boldsymbol{e}_h(\boldsymbol{\gamma}_s) \triangleq \boldsymbol{d} - \boldsymbol{g}(\boldsymbol{\gamma}_s) , \quad (51.92)$$

and the corresponding LS criterion is given by

$$J_h = \boldsymbol{e}_h^T \boldsymbol{e}_h = [\boldsymbol{d} - \boldsymbol{g}(\boldsymbol{\gamma}_s)]^T [\boldsymbol{d} - \boldsymbol{g}(\boldsymbol{\gamma}_s)] . \quad (51.93)$$

In the source localization problem, an observed range difference d_{n1} defines a hyperboloid in 3-D space. All points lying on such a hyperboloid are potential source locations and all have the same range difference d_{n1} to the two microphones n and 1. Therefore, a sound source that is located by minimizing (51.93) has the shortest distance to all hyperboloids associated with different microphone pairs and specified by the estimated range differences, after which fact the error criterion (51.93) is termed the hyperbolic error criterion (subscript 'h').

In (51.92), the signal model $\boldsymbol{g}(\boldsymbol{\gamma}_s)$ consists of a set of hyperbolic functions. Since they are nonlinear, minimizing (51.93) leads to a mathematically intractable solution as N becomes large. Moreover, the hyperbolic function is very sensitive to noise, especially for far-field sources. As a result, it is rarely used in practice.

When the statistical characteristics of the corrupting noise are unknown, uncorrelated *white* Gaussian noise is one reasonable assumption. In this case, it is not surprising that the hyperbolic LSE and the MLE minimize (maximize) similar criteria.

Spherical LS Error Function

The second LS criterion is based on the errors found in the distances from a hypothesized source location to the microphones. In the absence of measurement errors, the correct source location is preferably at the intersection of a group of spheres centered at the microphones. When

measurement errors are present, the best estimate of the source location would be the point that yields the shortest distance to those spheres defined by the range differences and the hypothesized source range.

Consider the distance D_n from the n-th microphone to the source. From the definition of the range difference (51.78) and the fact that $D_1 = \Omega_s$, we have:

$$\hat{D}_n = \Omega_s + d_{n1} \,, \tag{51.94}$$

where $\hat{D}_n$ denotes an observation based on the measured range difference. From the inner product, we can derive the true value for D_n^2, the square of the noise-free distance generated by a spherical signal model:

$$D_n^2 = \|\boldsymbol{\gamma}_n - \boldsymbol{\gamma}_s\|^2 = \Omega_n^2 - 2\boldsymbol{\gamma}_n^T\boldsymbol{\gamma}_s + \Omega_s^2 \,. \tag{51.95}$$

The spherical (subscript 'sp') LS error function is then defined as the difference between the measured and hypothesized values

$$\begin{aligned} e_{sp,n}(\boldsymbol{\gamma}_s) &\triangleq \frac{1}{2}\left(\hat{D}_n^2 - D_n^2\right) \\ &= \boldsymbol{\gamma}_n^T\boldsymbol{\gamma}_s + d_{n1}\Omega_s - \frac{1}{2}(\Omega_n^2 - d_{n1}^2) \,, \\ n &= 2, 3, \cdots, N \,. \end{aligned} \tag{51.96}$$

Putting the $N-1$ errors together and writing them in vector form gives,

$$\boldsymbol{e}_{sp}(\boldsymbol{r}_s) = \boldsymbol{A}\boldsymbol{\theta} - \boldsymbol{\xi} \,, \tag{51.97}$$

where

$$\boldsymbol{A} \triangleq \left[\boldsymbol{S} \,|\, \boldsymbol{d}\right], \quad \boldsymbol{S} \triangleq \begin{pmatrix} x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ & \vdots & \\ x_N & y_N & z_N \end{pmatrix},$$

$$\boldsymbol{\theta} \triangleq \begin{pmatrix} x_s \\ y_s \\ z_s \\ \Omega_s \end{pmatrix}, \quad \boldsymbol{\xi} \triangleq \frac{1}{2}\begin{pmatrix} \Omega_2^2 - d_{21}^2 \\ \Omega_3^2 - d_{21}^2 \\ \vdots \\ \Omega_N^2 - d_{21}^2 \end{pmatrix},$$

and $[\boldsymbol{S} \,|\, \boldsymbol{d}]$ indicates that $\boldsymbol{S}$ and $\boldsymbol{d}$ are stacked side by side. The corresponding LS criterion is then given by:

$$J_{sp} = \boldsymbol{e}_{sp}^T\boldsymbol{e}_{sp} = [\boldsymbol{A}\boldsymbol{\theta} - \boldsymbol{\xi}]^T[\boldsymbol{A}\boldsymbol{\theta} - \boldsymbol{\xi}] \,. \tag{51.98}$$

In contrast to the hyperbolic error function (51.92), the spherical error function (51.97) is linear in $\boldsymbol{\gamma}_s$ given Ω_s and vice versa. Therefore, the computational complexity to find a solution will *not* dramatically increase as N gets large.

51.3.6 Spherical Intersection (SX) Estimator

The SX source location estimator employs the spherical error and solves the problem in two steps [51.53]. It first finds the least-squares solution for $\boldsymbol{\gamma}_s$ in terms of Ω_s,

$$\boldsymbol{\gamma}_s = \boldsymbol{S}^\dagger(\boldsymbol{\xi} - \Omega_s\boldsymbol{d}) \,, \tag{51.99}$$

where $\boldsymbol{S}^\dagger \triangleq \left(\boldsymbol{S}^T\boldsymbol{S}\right)^{-1}\boldsymbol{S}^T$ is the pseudo-inverse of the matrix $\boldsymbol{S}$. Then, substituting (51.99) into the constraint $\Omega_s^2 = \boldsymbol{\gamma}_s^T\boldsymbol{\gamma}_s$ yields a quadratic equation as follows

$$\Omega_s^2 = \left[\boldsymbol{S}^\dagger\,(\boldsymbol{\xi} - \Omega_s\boldsymbol{d})\right]^T\left[\boldsymbol{S}^\dagger\,(\boldsymbol{\xi} - \Omega_s\boldsymbol{d})\right] \,. \tag{51.100}$$

After expansion, this becomes

$$\alpha_1\Omega_s^2 + \alpha_2\Omega_s + \alpha_3 = 0 \,, \tag{51.101}$$

where

$$\begin{aligned} &\alpha_1 = 1 - \|\boldsymbol{S}^\dagger\boldsymbol{d}\|^2, \quad \alpha_2 = 2\boldsymbol{\xi}^T\boldsymbol{S}^{\dagger T}\boldsymbol{S}^\dagger\boldsymbol{d} \,, \\ &\alpha_3 = -\|\boldsymbol{S}^\dagger\boldsymbol{\xi}\|^2 \,. \end{aligned}$$

The valid (real, positive) root is taken as an estimate of the source range Ω_s and is then substituted into (51.99) to calculate the SX estimate $\hat{\boldsymbol{\gamma}}_s^{SX}$ of the source location.

In the SX estimation procedure, the solution of the quadratic equation (51.101) for the source range Ω_s is required. This solution must be a positive value. If a real positive root is not available, the SX solution does not *exist*. On the contrary, if both of the roots are real and greater than 0, then the SX solution is not *unique*. In both cases, the SX source location estimator fails to produce a reliable estimate, which is not desirable for a real-time implementation.

51.3.7 Spherical Interpolation (SI) Estimator

In order to overcome this drawback of the SX algorithm, a spherical interpolation estimator was proposed in [51.54], which attempts to relax the restriction $\Omega_s = \|\boldsymbol{\gamma}_s\|$ by estimating Ω_s in the least-squares sense.

To begin, we substitute the least-squares solution (51.99) into the original spherical equation $\boldsymbol{A}\boldsymbol{\theta} = \boldsymbol{\xi}$ to obtain

$$\Omega_s\boldsymbol{P}_{\boldsymbol{S}^\perp}\boldsymbol{d} = \boldsymbol{P}_{\boldsymbol{S}^\perp}\boldsymbol{\xi} \,, \tag{51.102}$$

where

$$\boldsymbol{P}_{\boldsymbol{S}^\perp} \triangleq \boldsymbol{I}_{N\times N} - \boldsymbol{S}\boldsymbol{S}^\dagger \,, \tag{51.103}$$

and $\boldsymbol{I}_{N\times N}$ is the $N\times N$ identity matrix. The matrix $\boldsymbol{P}_{\boldsymbol{S}^\perp}$ is a projection matrix that projects a vector, when multiplied by the matrix, onto a space that is orthogonal

to the column space of $\boldsymbol{S}$. Such a projection matrix is symmetric (i. e., $\boldsymbol{P}_{\boldsymbol{S}^\perp} = \boldsymbol{P}_{\boldsymbol{S}^\perp}^{\mathrm{T}}$) and idempotent (i. e., $\boldsymbol{P}_{\boldsymbol{S}^\perp} = \boldsymbol{P}_{\boldsymbol{S}^\perp} \cdot \boldsymbol{P}_{\boldsymbol{S}^\perp}$). Then the least-squares solution to (51.102) is given by

$$\hat{\Omega}_{\mathrm{s}}^{\mathrm{SI}} = \frac{\boldsymbol{d}^{\mathrm{T}} \boldsymbol{P}_{\boldsymbol{S}^\perp} \boldsymbol{\xi}}{\boldsymbol{d}^{\mathrm{T}} \boldsymbol{P}_{\boldsymbol{S}^\perp} \boldsymbol{d}} \,. \tag{51.104}$$

Substituting this solution into (51.99) yields the SI estimate

$$\hat{\boldsymbol{\gamma}}_{\mathrm{s}}^{\mathrm{SI}} = \boldsymbol{S}^{\dagger} \left[\boldsymbol{I}_{N\times N} - \left(\frac{\boldsymbol{d}\boldsymbol{d}^{\mathrm{T}} \boldsymbol{P}_{\boldsymbol{S}^\perp}}{\boldsymbol{d}^{\mathrm{T}} \boldsymbol{P}_{\boldsymbol{S}^\perp} \boldsymbol{d}} \right) \right] \boldsymbol{\xi} \,. \tag{51.105}$$

In practice, the SI estimator performs better, but is computationally slightly more complex, than the SX estimator.

51.3.8 Linear-Correction Least-Squares Estimator

Finding the LSE based on the spherical error criterion (51.98) is a linear minimization problem, i. e.,

$$\hat{\boldsymbol{\theta}}^{\mathrm{LSE}} = \arg\min_{\boldsymbol{\theta}} (\boldsymbol{A}\boldsymbol{\theta} - \boldsymbol{\xi})^{\mathrm{T}} (\boldsymbol{A}\boldsymbol{\theta} - \boldsymbol{\xi}) \tag{51.106}$$

subject to a quadratic constraint

$$\boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\Xi} \boldsymbol{\theta} = 0 \,, \tag{51.107}$$

where $\boldsymbol{\Xi} \triangleq \mathrm{diag}(1, 1, 1, -1)$ is a diagonal and orthonormal matrix.

For such a constrained minimization problem, the technique of Lagrange multipliers will be used and the source location is determined by minimizing the Lagrangian

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}, \kappa) &= J_{\mathrm{sp}} + \kappa \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\Xi} \boldsymbol{\theta} \\ &= (\boldsymbol{A}\boldsymbol{\theta} - \boldsymbol{\xi})^{\mathrm{T}} (\boldsymbol{A}\boldsymbol{\theta} - \boldsymbol{\xi}) + \kappa \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\Xi} \boldsymbol{\theta} \,, \end{aligned} \tag{51.108}$$

where κ is a Lagrange multiplier. Expanding this expression yields

$$\mathcal{L}(\boldsymbol{\theta}, \kappa) = \boldsymbol{\theta}^{\mathrm{T}} \left(\boldsymbol{A}^{\mathrm{T}} \boldsymbol{A} + \kappa \boldsymbol{\Xi} \right) \boldsymbol{\theta} - 2 \boldsymbol{\xi}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{\theta} + \boldsymbol{\xi}^{\mathrm{T}} \boldsymbol{\xi} \,. \tag{51.109}$$

The necessary conditions for minimizing (51.109) can be obtained by taking the gradient of $\mathcal{L}(\boldsymbol{\theta}, \kappa)$ with respect to $\boldsymbol{\theta}$ and equating the result to zero. This produces:

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta}, \kappa)}{\partial \boldsymbol{\theta}} = 2 \left(\boldsymbol{A}^{\mathrm{T}} \boldsymbol{A} + \kappa \boldsymbol{\Xi} \right) \boldsymbol{\theta} - 2 \boldsymbol{A}^{\mathrm{T}} \boldsymbol{\xi} = \boldsymbol{0} \,. \tag{51.110}$$

Solving (51.110) for $\boldsymbol{\theta}$ yields the constrained least-squares estimate

$$\hat{\boldsymbol{\theta}} = \left(\boldsymbol{A}^{\mathrm{T}} \boldsymbol{A} + \kappa \boldsymbol{\Xi} \right)^{-1} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{\xi} \,, \tag{51.111}$$

where κ is yet to be determined.

In order to find κ, we can impose the quadratic constraint directly by substituting (51.111) into (51.107), which leads to

$$\boldsymbol{\xi}^{\mathrm{T}} \boldsymbol{A} \left(\boldsymbol{A}^{\mathrm{T}} \boldsymbol{A} + \kappa \boldsymbol{\Xi} \right)^{-1} \boldsymbol{\Xi} \left(\boldsymbol{A}^{\mathrm{T}} \boldsymbol{A} + \kappa \boldsymbol{\Xi} \right)^{-1} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{\xi} = 0 \,. \tag{51.112}$$

With eigenvalue analysis, the matrix $\boldsymbol{A}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{\Xi}$ can be decomposed as

$$\boldsymbol{A}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{\Xi} = \boldsymbol{U} \boldsymbol{\Lambda} \boldsymbol{U}^{-1} \,, \tag{51.113}$$

where $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \cdots, \lambda_4)$ and $\lambda_i, i = 1, \cdots, 4$, are the eigenvalues of the matrix $\boldsymbol{A}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{\Xi}$. Substituting (51.113) into (51.112), we may rewrite the constraint as:

$$\boldsymbol{p}^{\mathrm{T}} (\boldsymbol{\Lambda} + \kappa \boldsymbol{I})^{-2} \boldsymbol{q} = 0, \tag{51.114}$$

where

$$\begin{aligned} \boldsymbol{p} &= \boldsymbol{U}^{\mathrm{T}} \boldsymbol{\Xi} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{\xi}, \\ \boldsymbol{q} &= \boldsymbol{U}^{\mathrm{T}} \boldsymbol{A}^{\mathrm{T}} \boldsymbol{\xi} \,. \end{aligned}$$

Let us define a function of the Lagrange multiplier as follows

$$f(\kappa) \triangleq \boldsymbol{p}^{\mathrm{T}} (\boldsymbol{\Lambda} + \kappa \boldsymbol{I})^{-2} \boldsymbol{q} = \sum_{i=1}^{4} \frac{p_i q_i}{(\kappa + \lambda_i)^2} \,. \tag{51.115}$$

This is a polynomial of degree eight and, because of its complexity, numerical methods need to be used for root searching. Since the root of (51.115) for κ is not unique, a two-step procedure will be followed such that the desired source location could be found.

Unconstrained Spherical Least-Squares Estimator

In the first step, we assume that x_{s}, y_{s}, z_{s}, and Ω_{s} are mutually independent or equivalently we disregard the quadratic constraint (51.107) intentionally. Then the LS solution minimizing (51.98) for $\boldsymbol{\theta}$ (the source location as well as its range) is given by

$$\hat{\boldsymbol{\theta}}_1 = \boldsymbol{A}^{\dagger} \boldsymbol{\xi} \,, \tag{51.116}$$

where $\boldsymbol{A}^{\dagger} = \left(\boldsymbol{A}^{\mathrm{T}} \boldsymbol{A} \right)^{-1} \boldsymbol{A}^{\mathrm{T}}$ is the pseudo-inverse of the matrix $\boldsymbol{A}$.

A good parameter estimator first and foremost needs to be unbiased. For such an unconstrained spherical least-squares estimator, the bias and covariance matrix can be approximated by using the following perturbation analysis method.

When measurement errors are present in the range differences, $\boldsymbol{A}$, $\boldsymbol{\xi}$, and the parameter estimate $\hat{\boldsymbol{\theta}}_1$ deviate from their true values and can be expressed as:

$$\boldsymbol{A} = \boldsymbol{A}^{\mathrm{t}} + \Delta\boldsymbol{A}, \quad \boldsymbol{\xi} = \boldsymbol{\xi}^{\mathrm{t}} + \Delta\boldsymbol{\xi}, \quad \hat{\boldsymbol{\theta}}_1 = \boldsymbol{\theta}^{\mathrm{t}} + \Delta\boldsymbol{\theta}\,, \tag{51.117}$$

where variables with superscript 't' denote the true values, which also satisfy

$$\boldsymbol{\theta}^{\mathrm{t}} = \boldsymbol{A}^{\mathrm{t}\dagger}\boldsymbol{\xi}^{\mathrm{t}}\,. \tag{51.118}$$

If the magnitudes of the perturbations are small, the second-order errors are insignificant compared to their first-order counterparts and therefore can be neglected for simplicity, which then yields:

$$\Delta\boldsymbol{A} = [\boldsymbol{0}\ \ \boldsymbol{\epsilon}]\,, \quad \Delta\boldsymbol{\xi} \approx -\boldsymbol{d}^{\mathrm{t}} \odot \boldsymbol{\epsilon}\,, \tag{51.119}$$

where $\odot$ denotes the Schur (element-by-element) product. Substituting (51.117) into (51.116) gives,

$$\begin{aligned}&\left(\boldsymbol{A}^{\mathrm{t}} + \Delta\boldsymbol{A}\right)^{\mathrm{T}}\left(\boldsymbol{A}^{\mathrm{t}} + \Delta\boldsymbol{A}\right)\left(\boldsymbol{\theta}^{\mathrm{t}} + \Delta\boldsymbol{\theta}\right)\\ &= \left(\boldsymbol{A}^{\mathrm{t}} + \Delta\boldsymbol{A}\right)^{\mathrm{T}}\left(\boldsymbol{\xi}^{\mathrm{t}} + \Delta\boldsymbol{\xi}\right)\,.\end{aligned} \tag{51.120}$$

Retaining only the linear perturbation terms and using (51.118) and (51.119) produces:

$$\Delta\boldsymbol{\theta} \approx -\boldsymbol{A}^{\mathrm{t}\dagger}\boldsymbol{D}\boldsymbol{\epsilon}, \tag{51.121}$$

where $\boldsymbol{D} \triangleq \mathrm{diag}(D_2, D_3, \cdots, D_N)$ is a diagonal matrix. Since the measurement error $\boldsymbol{\epsilon}$ in the range differences has zero mean, $\hat{\boldsymbol{\theta}}_1$ is an unbiased estimate of $\boldsymbol{\theta}^{\mathrm{t}}$ when the small-error assumption holds:

$$E\{\Delta\boldsymbol{\theta}\} \approx E\left\{-\boldsymbol{A}^{t\,\dagger}\boldsymbol{D}\boldsymbol{\epsilon}\right\} = \boldsymbol{0}_{4\times 1}\,. \tag{51.122}$$

The covariance matrix of $\Delta\boldsymbol{\theta}$ is then found as,

$$\boldsymbol{R}_{\Delta\theta\Delta\theta} = E\left\{\Delta\boldsymbol{\theta}\Delta\boldsymbol{\theta}^{\mathrm{T}}\right\} = \boldsymbol{A}^{\mathrm{t}\dagger}\boldsymbol{D}\boldsymbol{R}_{\epsilon\epsilon}\boldsymbol{D}\boldsymbol{A}^{\mathrm{t}\dagger^{\mathrm{T}}}\,, \tag{51.123}$$

where $\boldsymbol{R}_{\epsilon\epsilon}$ is known or is properly assumed a priori. Theoretically, the covariance matrix $\boldsymbol{R}_{\Delta\theta\Delta\theta}$ cannot be calculated since it contains true values. Nevertheless, it can be approximated by using the values in $\hat{\boldsymbol{\theta}}_1$ with sufficient accuracy, as suggested by numerical studies.

In the first unconstrained spherical LS estimate (51.116), the range information is redundant because of the independence assumption on the source location and range. If that information is simply discarded, the source location estimate is the same as the SI estimate but with less computational complexity [51.55]. To demonstrate this, we first write (51.116) in block form as

$$\hat{\boldsymbol{\theta}}_1 = \begin{pmatrix}\boldsymbol{S}^{\mathrm{T}}\boldsymbol{S} & \boldsymbol{S}^{\mathrm{T}}\boldsymbol{d}\\ \boldsymbol{d}^{\mathrm{T}}\boldsymbol{S} & \boldsymbol{d}^{\mathrm{T}}\boldsymbol{d}\end{pmatrix}^{-1}\begin{pmatrix}\boldsymbol{S}^{\mathrm{T}}\\ \boldsymbol{d}^{\mathrm{T}}\end{pmatrix}\boldsymbol{\xi}\,. \tag{51.124}$$

It can easily be shown that:

$$\begin{pmatrix}\boldsymbol{S}^{\mathrm{T}}\boldsymbol{S} & \boldsymbol{S}^{\mathrm{T}}\boldsymbol{d}\\ \boldsymbol{d}^{\mathrm{T}}\boldsymbol{S} & \boldsymbol{d}^{\mathrm{T}}\boldsymbol{d}\end{pmatrix}^{-1} = \begin{pmatrix}\boldsymbol{Q} & \boldsymbol{v}\\ \boldsymbol{v}^{\mathrm{T}} & \varrho\end{pmatrix}\,, \tag{51.125}$$

where

$$\begin{aligned}\boldsymbol{v} &= -\left(\boldsymbol{S}^{\mathrm{T}}\boldsymbol{S} - \frac{\boldsymbol{S}^{\mathrm{T}}\boldsymbol{d}\boldsymbol{d}^{\mathrm{T}}\boldsymbol{S}}{\boldsymbol{d}^{\mathrm{T}}\boldsymbol{d}}\right)^{-1}\frac{\boldsymbol{S}^{\mathrm{T}}\boldsymbol{d}}{\boldsymbol{d}^{\mathrm{T}}\boldsymbol{d}}\,,\\ \boldsymbol{Q} &= \left(\boldsymbol{S}^{\mathrm{T}}\boldsymbol{S}\right)^{-1}\left[\boldsymbol{I} - \left(\boldsymbol{S}^{\mathrm{T}}\boldsymbol{d}\right)\boldsymbol{v}^{\mathrm{T}}\right]\,,\\ \varrho &= \frac{1 - (\boldsymbol{d}^{\mathrm{T}}\boldsymbol{S})\boldsymbol{v}}{\boldsymbol{d}^{\mathrm{T}}\boldsymbol{d}}\,.\end{aligned}$$

Next, we define another projection matrix $\boldsymbol{P}_{\boldsymbol{d}^{\perp}}$ associated with the $\boldsymbol{d}$-orthogonal space:

$$\boldsymbol{P}_{\boldsymbol{d}^{\perp}} \triangleq \boldsymbol{I} - \frac{\boldsymbol{d}\boldsymbol{d}^{\mathrm{T}}}{\boldsymbol{d}^{\mathrm{T}}\boldsymbol{d}}\,, \tag{51.126}$$

and find

$$\boldsymbol{v} = -\left(\boldsymbol{S}^{\mathrm{T}}\boldsymbol{P}_{\boldsymbol{d}^{\perp}}\boldsymbol{S}\right)^{-1}\frac{\boldsymbol{S}^{\mathrm{T}}\boldsymbol{d}}{\boldsymbol{d}^{\mathrm{T}}\boldsymbol{d}}\,, \tag{51.127}$$

$$\boldsymbol{Q} = \left(\boldsymbol{S}^{\mathrm{T}}\boldsymbol{P}_{\boldsymbol{d}^{\perp}}\boldsymbol{S}\right)^{-1}\,. \tag{51.128}$$

Substituting (51.125) together with (51.127) and (51.128) into (51.124) yields the unconstrained spherical LS estimate for the source coordinates:

$$\hat{\boldsymbol{\gamma}}_{\mathrm{s},1} = \left(\boldsymbol{S}^{\mathrm{T}}\boldsymbol{P}_{\boldsymbol{d}^{\perp}}\boldsymbol{S}\right)^{-1}\boldsymbol{S}^{\mathrm{T}}\boldsymbol{P}_{\boldsymbol{d}^{\perp}}\boldsymbol{\xi}\,, \tag{51.129}$$

which is the minimizer of

$$J_1(\boldsymbol{\gamma}_{\mathrm{s}}) = \left\|\boldsymbol{P}_{\boldsymbol{d}^{\perp}}\boldsymbol{\xi} - \boldsymbol{P}_{\boldsymbol{d}^{\perp}}\boldsymbol{S}\boldsymbol{\gamma}_{\mathrm{s}}\right\|^2\,, \tag{51.130}$$

or the least-squares solution to the linear equation

$$\boldsymbol{P}_{\boldsymbol{d}^{\perp}}\boldsymbol{S}\boldsymbol{\gamma}_{\mathrm{s}} = \boldsymbol{P}_{\boldsymbol{d}^{\perp}}\boldsymbol{\xi}. \tag{51.131}$$

In fact, the first unconstrained spherical LS estimator tries to approximate the projection of the observation vector $\boldsymbol{\xi}$ with the projections of the column vectors of the microphone location matrix $\boldsymbol{S}$ onto the $\boldsymbol{d}$-orthogonal space. The source location estimate is the coefficient vector associated with the *best* approximation. Clearly from (51.131), this estimation procedure is the generalization of the plane intersection (PI) method proposed in [51.56].

By using the Sherman–Morrison formula [51.57]

$$\left(\boldsymbol{A} + \boldsymbol{x}\boldsymbol{y}^{\mathrm{T}}\right)^{-1} = \boldsymbol{A}^{-1} - \frac{\boldsymbol{A}^{-1}\boldsymbol{x}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{A}^{-1}}{1 + \boldsymbol{y}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{x}}\,, \tag{51.132}$$

we can expand the item in (51.129) as

$$\left(\boldsymbol{S}^{\mathrm{T}}\boldsymbol{P}_{\boldsymbol{d}^{\perp}}\boldsymbol{S}\right)^{-1}=\left[\boldsymbol{S}^{\mathrm{T}}\boldsymbol{S}-\left(\frac{\boldsymbol{S}^{\mathrm{T}}\boldsymbol{d}}{\boldsymbol{d}^{\mathrm{T}}\boldsymbol{d}}\right)(\boldsymbol{S}^{\mathrm{T}}\boldsymbol{d})^{\mathrm{T}}\right]^{-1},$$

and finally show that the unconstrained spherical LS estimate (51.129) is equivalent to the SI estimate (51.105), i. e., $\hat{\boldsymbol{\gamma}}_{\mathrm{s},1}\equiv\hat{\boldsymbol{\gamma}}_{\mathrm{s}}^{\mathrm{SI}}$.

Although the unconstrained spherical LS and the SI estimators are mathematically equivalent, they are quite different in computational efficiency due to their different approaches to the source localization problem. The complexities of the SI and unconstrained spherical LS estimators are $\mathcal{O}\left(N^3\right)$ and $\mathcal{O}(N)$, respectively. In comparison, the unconstrained spherical LS estimator reduces the complexity of the SI estimator by a factor of N^2, which is significant when N is large (more microphones are used).

Linear Correction

In the previous subsection, we developed the unconstrained spherical LS estimator (USLSE) for source localization and demonstrated that it is mathematically equivalent to the SI estimator but with less computational complexity. Although the USLSE/SI estimates can be accurate, as indicated in [51.55] among others, it is helpful to exploit the redundancy of source range to improve the statistical efficiency (i. e., to reduce the variance of source location estimates) of the overall estimation procedure. Therefore, in the second step, we intend to correct the USLS estimate $\hat{\boldsymbol{\theta}}_1$ to make a better estimate $\hat{\boldsymbol{\theta}}_2$ of $\boldsymbol{\theta}$. This new estimate should be in the neighborhood of $\hat{\boldsymbol{\theta}}_1$ and should obey the constraint (51.107). We expect that the corrected estimate will still be unbiased and have a smaller variance.

To begin, we substitute $\hat{\boldsymbol{\theta}}_1=\boldsymbol{\theta}^{\mathrm{t}}+\Delta\boldsymbol{\theta}$ into (51.110) and expand the expression to find

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\hat{\boldsymbol{\theta}}_1+\kappa\boldsymbol{\Xi}\hat{\boldsymbol{\theta}}_1-(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\kappa\boldsymbol{\Xi})\Delta\boldsymbol{\theta}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{\xi}\,. \tag{51.133}$$

Combined with (51.116), (51.133) becomes

$$(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\kappa\boldsymbol{\Xi})\Delta\boldsymbol{\theta}=\kappa\boldsymbol{\Xi}\hat{\boldsymbol{\theta}}_1\,, \tag{51.134}$$

and hence

$$\Delta\boldsymbol{\theta}=\kappa\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{\Xi}\boldsymbol{\theta}^{\mathrm{t}}\,. \tag{51.135}$$

Substituting (51.135) into $\hat{\boldsymbol{\theta}}_1=\boldsymbol{\theta}^{\mathrm{t}}+\Delta\boldsymbol{\theta}$ yields

$$\hat{\boldsymbol{\theta}}_1=\left[\boldsymbol{I}+\kappa\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{\Xi}\right]\boldsymbol{\theta}^{\mathrm{t}}\,. \tag{51.136}$$

Solving (51.136) for $\boldsymbol{\theta}^{\mathrm{t}}$ produces the corrected estimate $\hat{\boldsymbol{\theta}}_2$ and also the final output of the linear-correction least-squares (LCLS) estimator:

$$\hat{\boldsymbol{\theta}}_2=\left[\boldsymbol{I}+\kappa\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{\Xi}\right]^{-1}\hat{\boldsymbol{\theta}}_1\,. \tag{51.137}$$

Equation (51.137) suggests how the second-step processing updates the source location estimate based on the first unconstrained spherical least squares result, or equivalently the SI estimate. If the regularity condition [51.58]

$$\lim_{i\to\infty}\left[\kappa(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{\Xi}\right]^{i}=\boldsymbol{0} \tag{51.138}$$

is satisfied, then the estimate $\hat{\boldsymbol{\theta}}_2$ can be expanded in a Neumann series :

$$\begin{aligned}\hat{\boldsymbol{\theta}}_2&=\left\{\boldsymbol{I}+\left[-\kappa\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{\Xi}\right]\right.\\&\quad\left.+\left[-\kappa\left[\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right]^{-1}\boldsymbol{\Xi}\right]^{2}+\cdots\right\}\hat{\boldsymbol{\theta}}_1\\&=\hat{\boldsymbol{\theta}}_1+\sum_{i=1}^{\infty}\left[-\kappa\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{\Xi}\right]^{i}\hat{\boldsymbol{\theta}}_1\,,\end{aligned} \tag{51.139}$$

where the second term is the linear correction. Equation (51.138) implies that, in order to avoid divergence, the Lagrange multiplier κ should be small. In addition, κ needs to be determined carefully such that $\hat{\boldsymbol{\theta}}_2$ obeys the quadratic constraint (51.107).

Because the function $f(\kappa)$ is smooth near $\kappa=0$ (corresponding to the neighborhood of $\hat{\boldsymbol{\theta}}_1$), as suggested by numerical experiments, the secant method [51.59] can be used to determine its desired root. Two reasonable initial points can be chosen as:

$$\kappa_0=0,\quad \kappa_1=\beta\,, \tag{51.140}$$

where the small number β depends on the array geometry. Five iterations should be sufficient to give an accurate approximation to the root.

The idea of exploiting the relationship between a sound source's range and its location coordinates to improve the estimation efficiency of the SI estimator was first suggested by *Chan* and *Ho* [51.52] with a quadratic correction. Accordingly, they constructed a quadratic data model for $\hat{\boldsymbol{\theta}}_1$.

$$\hat{\boldsymbol{\theta}}_1\odot\hat{\boldsymbol{\theta}}_1=\boldsymbol{T}(\boldsymbol{\gamma}_{\mathrm{s}}\odot\boldsymbol{\gamma}_{\mathrm{s}})+\boldsymbol{v}\,, \tag{51.141}$$

where

$$\boldsymbol{T} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

is a constant matrix, and $\boldsymbol{n}$ is the corrupting noise. In contrast to the linear correction technique based on the Lagrange multiplier, the quadratic counterpart needs to know the covariance matrix $\boldsymbol{R}_{\epsilon\epsilon}$ of measurement errors in the range differences a priori. In a real-time digital source localization system, a poorly estimated $\boldsymbol{R}_{\epsilon\epsilon}$ will lead to performance degradation. In addition, the quadratic-correction least-squares estimation procedure uses the perturbation approaches to linearly approximate $\Delta\boldsymbol{\theta}$ and $\boldsymbol{v}$ in (51.117) and (51.141), respectively. Therefore, the approximations of their corresponding covariance matrices $\boldsymbol{R}_{\Delta\theta\Delta\theta}$ and $\boldsymbol{R}_{\nu\nu}$ can be good only when the noise level is low. When noise is at a practically high level, the quadratic-correction least squares estimate has a large bias and a high variance. Furthermore, since the true value of the source location that is necessary for calculating $\boldsymbol{R}_{\Delta\theta\Delta\theta}$ and $\boldsymbol{R}_{\nu\nu}$ cannot be known theoretically, the estimated source location has to be utilized for approximation. It was suggested in [51.52] that several iterations in the second correction stage would improve estimation accuracy. However, while the bias is suppressed after iterations, the estimate is closer to the SI solution and the variance is boosted, as reported in [51.50]. Finally, the direct solutions of the quadratic-correction least-squares estimator are the squares of the source location coordinates $\boldsymbol{\gamma}_s \odot \boldsymbol{\gamma}_s$. In 3-D space, these correspond to eight positions, which introduce decision ambiguities. Other physical criteria, such as the domain of interest, were suggested, but these are hard to define in practical situations, particularly when one of the source coordinates is close to zero.

In comparison, the linear-correction method updates the source location estimate of the first unconstrained spherical LS estimator without making any assumption about the error covariance matrix and without resorting to a linear approximation. Even though we need to find a small root of function (51.115) for the Lagrange multiplier κ that satisfies the regularity condition (51.138), the function $f(\kappa)$ is smooth around zero and the solution can easily be determined using the secant method. The linear-correction method achieves a better balance between computational complexity and estimation accuracy.

51.4 Summary

In this chapter, we have explained why TDE-based two-stage approaches have become the technique of choice for real-time speech source localization using microphone arrays. We presented a comprehensive overview of the research and development on this technology over the last three decades.

The chapter consists of two parts. The first part was devoted to the TDE problem. We began with a close examination of two signal models used in this problem. Their strengths and drawbacks were discussed. We reviewed the well-known generalized cross-correlation (GCC) methods and intended to treat this class of methods as a benchmark for a number of recently developed new algorithms. We made it clear by analysis that the GCC methods cannot cope well with room reverberation and described two strategies to combat this effect. One strategy is to employ the more-realistic real reverberant model in the formulation of the TDE problem and try to blindly identify the acoustic channel impulse responses from which the time difference of arrival (TDOA) is inferred. The adaptive eigenvalue decomposition algorithm and the adaptive blind multichannel identification algorithms were developed. The other strategy is to exploit the TDOA redundancy among different microphone pairs. This leads to the multichannel spatial predication and interpolation methods, the multichannel cross-correlation coefficient method, and the minimum-entropy method, which have all been thoroughly explored.

The second part of the Chapter surveyed the source localization techniques. The localization problem was postulated from the perspective of estimation theory, and the Cramèr–Rao lower bound for unbiased location estimators was derived. After an insightful review of conventional approaches ranging from maximum-likelihood to least-squares estimators, we presented a recently developed linear-correction least-squares algorithm that is more robust to measurement errors and that is computationally as well as statistically more efficient.

References

51.1 D.R. Fischell, C.H. Coker: A speech direction finder, Proc. ICASSP (1984) pp. 19.8.1–19.8.4.

51.2 H.F. Silverman: Some analysis of microphone arrays for speech data analysis, IEEE Trans. ASSP **35**, 1699–1712 (1987)

51.3 J.L. Flanagan, A. Surendran, E. Jan: Spatially selective sound capture for speech and audio processing, Speech Commun. **13**, 207–222 (1993)

51.4 D.B. Ward, G.W. Elko: Mixed nearfield/farfield beamforming: a new technique for speech acquisition in a reverberant environment, Proc. IEEE ASSP Workshop Appl. Signal Process. Audio Acoust. (1997)

51.5 D.V. Rabinkin, R.J. Ranomeron, J.C. French, J.L. Flanagan: A DSP implementation of source location using microphone arrays, Proc. SPIE **2846**, 88–99 (1996)

51.6 H. Wang, P. Chu: Voice source localization for automatic camera pointing system in videoconferencing, Proc. IEEE ASSP Workshop Appl. Signal Process. Audio Acoust. (1997)

51.7 C. Wang, M.S. Brandstein: A hybrid real-time face tracking system, Proc. ICASSP, Vol. 6 (1998) pp. 3737–3741

51.8 Y. Huang, J. Benesty, G.W. Elko: Microphone arrays for video camera steering. In: *Acoustic Signal Processing for Telecommunication*, ed. by S.L. Gay, J. Benesty (Kluwer Academic, Boston 2000) pp. 239–259, chap. 11

51.9 S. Haykin: Radar array processing for angle of arrival estimation. In: *Array Signal Process*, ed. by S. Haykin (Prentice-Hall, Englewood Cliffs 1985)

51.10 H. Krim, M. Viberg: Two decades of array signal processing research: the parametric approach, IEEE Signal Process. Mag. **13**(4), 67–94 (1996)

51.11 R.J. Vaccaro: The past, present; future of underwater acoustic signal processing, IEEE Signal Process. Mag. **15**, 21–51 (1998)

51.12 D.V. Sidorovich, A.B. Gershman: Two-dimensional wideband interpolated root-MUSIC applied to measured seismic data, IEEE Trans. Signal Process. **46**(8), 2263–2267 (1998)

51.13 J. Capon: Maximum-likelihood spectral estimation, Proc. IEEE **57**, 1408–1418 (1969)

51.14 R.O. Schmidt: *A Signal Subspace Approach to Multiple Emitter Location and Spectral Estimation* (Stanford University, Stanford 1981), Ph.D. thesis

51.15 W. Bangs, P. Schultheis: Space–time processing for optimal parameter estimation. In: *Signal Process*, ed. by J. Griffiths, P. Stocklin, C. Van Schooneveld (New York, Academic 1973) pp. 577–590

51.16 W.R. Hahn, S.A. Tretter: Optimum processing for delay-vector estimation in passive signal arrays, IEEE Trans. Inform. Theory **19**, 608–614 (1973)

51.17 M. Wax, T. Kailath: Optimum localization of multiple sources by passive arrays, IEEE Trans. ASSP **31**(5), 1210–1218 (1983)

51.18 M.S. Brandstein, H.F. Silverman: A practical methodology for speech source localization with microphone arrays, Comput. Speech Lang. **2**, 91–126 (1997)

51.19 C.H. Knapp, G.C. Carter: The generalized correlation method for estimation of time delay, IEEE Trans. ASSP **24**, 320–327 (1976)

51.20 G.C. Carter, A.H. Nuttall, P.G. Cable: The smoothed coherence transform, Proc. IEEE **61**, 1497–1498 (1973)

51.21 J.P. Ianniello: Time delay estimation via cross-correlation in the presence of large estimation errors, IEEE Trans. ASSP **30**, 998–1003 (1982)

51.22 B. Champagne, S. Bédard, A. Stéphenne: Performance of time-delay estimation in presence of room reverberation, IEEE Trans. Speech Audio Process. **4**, 148–152 (1996)

51.23 M. Omologo, P. Svaizer: Acoustic event localization using a crosspower-spectrum phase based technique, Proc. ICASSP, Vol. 2 (1994) pp. 273–276

51.24 M.S. Brandstein: A pitch-based approach to time-delay estimation of reverberant speech, Proc. IEEE ASSP Workshop Appl. Signal Process. Audio Acoustics (1997)

51.25 M. Omologo, P. Svaizer: Acoustic source location in noisy and reverberant environment using CSP analysis, ICASSP, Vol. 2 (1996) pp. 921–924

51.26 A. Stéphenne, B. Champagne: Cepstral prefiltering for time delay estimation in reverberant environments, Proc. ICASSP, Vol. 5. (1995) pp. 3055–3058

51.27 J. Benesty: Adaptive eigenvalue decomposition algorithm for passive acoustic source localization, J. Acoust. Soc. Am. **107**, 384–391 (2000)

51.28 G. Xu, H. Liu, L. Tong, T. Kailath: A least-squares approach to blind channel identification, IEEE Trans. Signal Process. **43**, 2982–2993 (1995)

51.29 Y. Huang, J. Benesty: Adaptive multichannel time delay estimation based on blind system identification for acoustic source localization. In: *Adaptive Signal Processing: Application to Real-World Problems*, ed. by J. Benesty, Y. Huang (Springer, Berlin:Heidelberg 2003)

51.30 Y. Huang, J. Benesty: Adaptive multi-channel least mean square and Newton algorithms for blind channel identification, Signal Process. **82**, 1127–1138 (2002)

51.31 Y. Huang, J. Benesty: A class of frequency-domain adaptive approaches to blind multichannel identification, IEEE Trans. Signal Process. **51**, 11–24 (2003)

51.32 Y. Huang, J. Benesty, J. Chen: Optimal step size of the adaptive multichannel LMS algorithm for blind

SIMO identification, IEEE Signal Process. Lett. **12**, 173–176 (2005)
51.33 Y. Huang, J. Benesty, J. Chen: *Acoustic MIMO Signal Process* (Berlin, Springer 2006)
51.34 R.L. Kirlin, D.F. Moore, R.F. Kubichek: Improvement of delay measurements from sonar arrays via sequential state estimation, IEEE Trans. ASSP **29**, 514–519 (1981)
51.35 T. Nishiura, T. Yamada, S. Nakamura, K. Shikano: Localization of multiple sound sources based on a CSP analysis with a microphone array, Proc. ICASSP (2000) pp. 1053–1055
51.36 S.M. Griebel, M.S. Brandstein: Microphone array source localization using realizable delay vectors, Proc. IEEE ASSP Workshop Appl. Signal Process. Audio Acoust. (2001) pp. 71–74
51.37 J. DiBiase, H. Silverman, M. Brandstein: Robust localization in reverberant rooms. In: *Microphone Arrays: Signal Processing Techniques and Applications*, ed. by M. Branstein, D. Ward (Springer, Berlin 2001)
51.38 J. Chen, J. Benesty, Y. Huang: Robust time delay estimation exploiting redundancy among multiple microphones, IEEE Trans. Speech Audio Process. **11**, 549–557 (2003)
51.39 J. Benesty, J. Chen, Y. Huang: Time-delay estimation via linear interpolation and cross-correlation, IEEE Trans. Speech Audio Process. **12**, 509–519 (2004)
51.40 J.S. Bendat, A.G. Piersol: *Random Data Analysis and Measurement Procedures* (Wiley, New York 1986)
51.41 D. Cochran, H. Gish, D. Sinno: A geometric approach to multichannel signal detection, IEEE Trans. Signal Process. **43**, 2049–2057 (1995)
51.42 C.E. Shannon: A mathematical theory of communication, Bell Syst. Tech. J. **27**, 379–423, 623–656 (1948)
51.43 T.M. Cover, J.A. Thomas: *Elements of Information Theory* (Wiley, New York 1991)
51.44 I. Kojadinovic: On the use of mutual information in data analysis: an overview, Int. Symposium on Applied Stochastic Models and Data Analysis (2005)
51.45 J. Benesty, Y. Huang, J. Chen: Time delay estimation via minimum entropy, IEEE Signal Process. Lett. **14**, 157–160 (2006)
51.46 L.R. Rabiner, R.W. Schafer: *Digital Process. of Speech Signals* (Prentice-Hall, Englewood Cliffs 1978)
51.47 S. Gazor, W. Zhang: Speech probability distribution, IEEE Signal Process. Lett. **10**(7), 204–207 (2003)
51.48 S. Kotz, T.J. Kozubowski, K. Podgórski: *An asymmetric multivariate Laplace distribution*, Technical Report No. 367, Department of Statistics and Applied Probaility (Univ. of California, Santa Barbara 2000)
51.49 T. Eltoft, T. Kim, T.-W. Lee: On the multivariate Laplace distribution, IEEE Signal Process. Lett. **13**, 300–303 (2006)
51.50 Y. Huang, J. Benesty, G.W. Elko, R.M. Mersereau: Real-time passive source localization: an unbiased linear-correction least-squares approach, IEEE Trans. Speech Audio Process. **9**, 943–956 (2001)
51.51 S.M. Kay: *Fundamentals of Statistical Signal Process.: Estimation Theory* (Prentice-Hall, Englewood Cliffs 1993)
51.52 Y.T. Chan, K.C. Ho: A simple and efficient estimator for hyperbolic location, IEEE Trans. Signal Process. **42**, 1905–1915 (1994)
51.53 H.C. Schau, A.Z. Robinson: Passive source localization employing intersecting spherical surfaces from time-of-arrival differences, IEEE Trans. ASSP **35**, 1223–1225 (1987)
51.54 J.S. Abel, J.O. Smith: The spherical interpolation method for closed-form passive source localization using range difference measurements echo cancelation, Proc. ICASSP (1987) pp. 471–474
51.55 Y. Huang, J. Benesty, G.W. Elko: Passive acoustic source localization for video camera steering, Proc. IEEE Int. Conf. ACSP, Vol. 2. (2000) pp. 909–912
51.56 R.O. Schmidt: A new approach to geometry of range difference location, IEEE Trans. Aerosp. Electron. **8**, 821–835 (1972)
51.57 T.K. Moon, W.C. Stirling: *Mathematical Methods and Algorithms* (Prentice-Hall, Upper Saddle River 1999)
51.58 C.D. Meyer: *Matrix Analysis and Applied Linear Algebra* (SIAM, Philadelphia 2000)
51.59 W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling: *Numerical Recipes in C: The Art of Scientific Computing* (Cambridge Univ. Press, Cambridge 1988)

Acknowledgements

C.16 Low-Bit-Rate Speech Coding
by A. V. McCree

This work was sponsored by the Department of Defense under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

D.25 Expressive/Affective Speech Synthesis
by N. Campbell

The model arose from extended discussions during and after SP2006, and the author wishes to acknowledge contributions from Sacha Fagel of the TU Berlin. This work is partly supported by the Ministry of Public Management, Home Affairs, Posts and Telecommunications, Japan under the SCOPE funding initiative. The ESP corpus was collected over a period of five years with support from the Japan Science & Technology Corporation core research for evolutional science & technology (JST/CREST) funding initiative. The paper was written while the author was employed by the National Institute of Information and Communications Technology. The author also wishes to thank the management of the Spoken Language Communication Research Laboratory and the Advanced Telecommunications Research Institute International for their continuing support and encouragement of this work.

E.26 Historical Perspective of the Field of ASR/NLU
by L. Rabiner, B.-H. Juang

The authors thank Dr. John Makhoul and Prof. Victor Zue for their comments and insights on the material presented in this paper.

E.28 Speech Recognition with Weighted Finite-State Transducers
by M. Mohri, F. Pereira, M. Riley

We thank Andrej Ljolje for providing the acoustic models, and Don Hindle and Richard Sproat for providing the language models used in our experiments.

The first author's work was partially funded by the New York State Office of Science Technology and Academic Research (NYSTAR). The second author's work was partly funded by NSF grants EIA 0205456, EIA 0205448, and IIS 0428193.

E.30 Towards Superhuman Speech Recognition
by M. Picheny, D. Nahamoo

We would like to thank Brian Kingsbury, Lidia Mangu, Mukund Padmanabhan, and Geoff Zweig for their contributions to the research and many of the ideas in this chapter, and the entire human language technologies group at the IBM TJ Watson Research Center for creating a truly *Superhuman* environment for speech recognition research.

E.35 Spoken Dialogue Systems
by V. Zue, S. Seneff

The research described in this chapter is the result of collaboration with many past and current students and staff of the Spoken Language Systems Group at the MIT Laboratory for Computer Science, especially Jim Glass. Their contributions are gratefully acknowledged. This research is sponsored by the T-Party Project, a joint research program between MIT and Quanta Computer Inc.

F.38 Text-Independent Speaker Recognition
by D. A. Reynolds, W. M. Campbell

This work was sponsored by the Department of Defense under Air Force contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States government.

G.41 Automatic Language Recognition Via Spectral and Token Based Approaches
by D. A. Reynolds, W. M. Campbell, W. Shen, E. Singer

This work was sponsored by the Department of Defense under Air Force contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States government.

I.52 Convolutive Blind Source Separation Methods
by M. S. Pedersen, J. Larsen, U. Kjems, L. C. Parra

M.S.P. was supported by the Oticon Foundation. M.S.P. and J.L. are partly also supported by the European Commission through the sixth framework IST Network of Excellence: pattern analysis, statistical modeling and computational learning (PASCAL).

About the Authors

Alex Acero

Microsoft Research
Redmond, WA, USA
alexac@microsoft.com

Chapter E.33

Dr. Acero is Research Area Manager at Microsoft Research, overseeing research in speech technology, natural language, computer vision, communication and multimedia collaboration. Dr. Acero is a Fellow of IEEE, VP for the IEEE Signal Processing Society and was Distinguished Lecturer. He is author of the two books, and over 140 technical papers. He holds 29 US patents. He obtained his Ph.D. from Carnegie Mellon, Pittsburgh, PA.

Jont B. Allen

University of Illinois
ECE
Urbana, IL, USA
JontAllen@ieee.org

Chapter A.3

Jont B. Allen received his BS in Electrical Engineering from the University of Illinois, Urbana-Champaign, in 1966, and his MS and Ph.D. in Electrical Engineering from the University of Pennsylvania in 1968 and 1970, respectively. After graduation he joined Bell Laboratories, and was in the Acoustics Research Department in Murray Hill NJ from 1974 to 1996, as a Distinguished member of Technical Staff. Since 1996 Dr. Allen was a Technology Leader at AT&T Labs-Research. Since Aug. 2003 Allen is an Associate Professor in ECE, at the University of Illinois, and on the research staff of the Beckman Inst., Urbana IL. During his 32 year AT&T career Prof. Allen has specialized in cochlear and middle ear modeling and auditory signal processing. In the last 10 years, he has concentrated on the problem of human speech recognition. His expertise spans the areas of signal processing, physical acoustics, acoustic power flow and impedance measurements, cochlear modeling, auditory neurophysiology, auditory psychophysics, and human speech recognition. Professor Allen is a Fellow (May 1981) of the Acoustical Society of America (ASA) and Fellow (January 1985) of the Institute of Electrical and Electronic Engineers (IEEE). In 1986 he was awarded the IEEE Acoustics Speech and Signal Processing (ASSP) Society Meritorious Service Award, and in 2000 received an IEEE Third Millennium Medal.

Jacob Benesty

University of Quebec
INRS-EMT
Montreal, Quebec, Canada
benesty@emt.inrs.ca

Chapters 1, B.6, B.7, B.13, H.43, H.46, I.51

Jacob Benesty received the Master's degree from Pierre & Marie Curie University, France, in 1987, and the Ph.D. degree in control and signal processing from Orsay University, France, in 1991. After positions at Telecom Paris University as a consultant and a member of the technical staff at Bell Laboratories, Murray Hill, Dr. Benesty joined the University of Québec (INRS-EMT) in Montreal, Canada in May 2003 as a professor. His research interests are in signal processing, acoustic signal processing, and multimedia communications.

Frédéric Bimbot

IRISA (CNRS & INRIA) – METISS
Rennes, France
bimbot@irisa.fr

Chapter F.36

After receiving his Ph.D. in 1988, Frédéric Bimbot joined CNRS in 1990 as a permanent researcher, working initially with ENST-Paris and then with IRISA in Rennes. He also repeatedly visited AT&T – Bell Laboratories between 1990 and 1999. His research is focused on audio signal analysis, speech modelling, speaker characterization and audio source separation. Since 2002, he is heading the METISS research group dedicated to selected topics in speech and audio processing.

Thomas Brand

Carl von Ossietzky Universität Oldenburg
Sektion Medizinphysik
Oldenburg, Germany
thomas.brand@uni-oldenburg.de

Chapter A.4

Dr. Thomas Brand studied physics in Göttingen and Oldenburg where he received his Ph.D. for his work on analysis and optimization of psychophysical procedures in audiology. He is now scientist at the University of Oldenburg, Germany, where his research activities focus on psychoacoustics in audiology and measurement and prediction of speech intelligibility.

Nick Campbell

Knowledge Creating Communication
Research Centre
Acoustics & Speech Research Project,
Spoken Language Communication Group
Keihanna Science City, Japan
nick@nict.go.jp

Chapter D.25

Nick Campbell received his Ph.D. in Experimental Psychology from the University of Sussex in the U.K. and is currently engaged as a Chief Researcher in the Department of Acoustics and Speech Research at the Advanced Telecommunications Research Institute International (ATR) in Kyoto, Japan, where he also serves as Research Director for the JST/CREST Expressive Speech Processing and the SCOPE *Robot's Ears* projects. He was first invited as a Research Fellow at the IBM UK Scientific Centre, where he developed algorithms for speech synthesis, and later at the AT&T Bell Laboratories where he worked on the synthesis of Japanese. He served as Senior Linguist at the Edinburgh University Centre for Speech Technology Research before joining ATR in 1990. His research interests are based on large speech databases and include nonverbal speech processing, concatenative speech synthesis, and prosodic information modelling. He spends his spare time working with postgraduate students as Visiting Professor at the Nara Institute of Science and Technology (NAIST) and at Kobe University in Japan.

William M. Campbell

MIT Lincoln Laboratory
Information Systems Technology Group
Lexington, MA, USA
wcampbell@ll.mit.edu

Chapters F.38, G.41

Dr. Campbell is a technical staff member in the Information Systems Technology group at MIT Lincoln Laboratory. He received his Ph.D. in Applied Mathematics from Cornell University in 1995. Prior to joining MIT Lincoln Laboratory, he worked at Motorola on biometrics, speech interfaces, wearable computing, and digital communications. His current research interests include machine learning, speech processing, and plan recognition.

Rolf Carlson

Royal Institute of Technology (KTH)
Department of Speech, Music and Hearing
Stockholm, Sweden
rolf@speech.kth.se

Chapter D.20

Rolf Carlson is Professor in Speech Technology at KTH since 1995. Early research together with Björn Granström resulted in a multi-lingual text-to-speech system in 1976. This formed the base of the speech technology company Infovox in 1982. He has served on the board of ISCA and has published numerous papers in the speech research and technology area, including work on multimodal spoken dialog systems.

Jingdong Chen

Bell Laboratories
Alcatel-Lucent
Murray Hill, NJ, USA
jingdong@research.bell-labs.com

Chapters B.6, B.7, B.13, H.43, H.46, I.51

Dr. Chen holds a Ph.D in pattern recognition and intelligence control, an M.S. and a B.S. in electrical engineering. From 1998 to 1999, he worked at ATR Interpreting Telecommunications Research Laboratories, Japan. From 1999 to 2000, he worked at Griffith University, Australia and from 2000 to 2001, he was with ATR Spoken Language Translation Research Laboratories. He joined Bell Laboratories in 2001 as a Member of Technical Staff. He has extensively published in the area of acoustic signal processing for speech communications and authored/edited two books. He has helped organizing several international conferences and workshops, and serves as a member of the IEEE Signal Processing Society Technical Committee on Audio & Electroacoustics.

Juin-Hwey Chen

Broadcom Corp.
Irvine, CA, USA
rchen@broadcom.com

Chapter C.17

Juin-Hwey Chen is a Senior Technical Director at Broadcom Corporation. He received his Ph.D. in Electrical Engineering from UC Santa Barbara in 1987. He is the primary inventor of the ITU-T G.728 speech coding standard and the BroadVoice speech coding standards in PacketCable, SCTE, ANSI, and ITU-T J.161. He was elected an IEEE Fellow in 1995 and a Broadcom Fellow in 2006.

Israel Cohen

Technion–Israel Institute of Technology
Department of Electrical Engineering
Haifa, Israel
icohen@ee.technion.ac.il

Chapters H.44, H.47

Dr. Israel Cohen is a Professor of Electrical Engineering at the Technion–Israel Institute of Technology. He obtained his Ph.D. degree in electrical engineering from the Technion in 1998. From 1990 to 1998, he was a Research Scientist with RAFAEL research laboratories, Israel Ministry of Defence. From 1998 to 2001, he was a Postdoctoral Research Associate with the Computer Science Department at Yale University. He is a senior member of IEEE.

Jordan Cohen

SRI International
Menlo Park, CA, USA
jrc@speech.sri.com

Chapter E.34

Jordan Cohen is a Senior Scientist at SRI International, specializing in language based applications. He is the Principal Investigator for the DARPA GALE program for SRI, coordinating the activities of 14 subcontractors to harness speech recognition, language translation, and information annotation and distillation resources for the government. Jordan received his Ph.D. in Linguistics from the University of Connecticut, preceeded by a Masters' Degree in Electrical Engineering from the University of Illinois. He has worked in the Department of Defense, IBM, and at the Institute for Defense Analyses in Princeon, NJ. Most recently, Jordan was the CTO of Voice Signal Technologies, a company which produces multimodal speech-centric interfaces for mobile devices. He is a member of the Acoustical Society of America and the IEEE, and has published in both the classified and unclassified literature of speech and language technologies.

Corinna Cortes

Google, Inc.
Google Research
New York, NY, USA
corinna@google.com

Chapter E.29

Corinna Cortes is the Head of Google Research, NY, where she is working on a broad range of theoretical and applied large-scale machine learning problems. Prior to Google, Corinna spent more than ten years at AT&T Labs - Research, formerly AT&T Bell Labs, where she held a distinguished research position. Corinna's research work is well-known in particular for her contributions to the theoretical foundations of support vector machines (SVMs) and her work on data-mining in very large data sets for which she was awarded the AT&T Science and Technology Medal in the year 2000. Corinna received her Ph.D. in computer science from the University of Rochester in 1993.

Eric J. Diethorn

Avaya Labs Research
Multimedia Technologies Research Department
Basking Ridge, NJ, USA
ejd@avaya.com

Chapter H.43

Dr. Diethorn received his Ph.D. in Electrical Engineering from the University of Illinois, Urbana-Champaign, in 1987. His professional focus is acoustical signal processing for speech communications, and he has extensive experience in the fields of acoustic- and electrical-echo cancellation, noise reduction and speech enhancement. Prior to joining Avaya in 2002, Dr. Diethorn worked for AT&T Bell Laboratories, Lucent Technologies, and Agere Systems.

Simon Doclo

Katholieke Universiteit Leuven
Department of Electrical Engineering
(ESAT-SCD)
Leuven, Belgium
simon.doclo@esat.kuleuven.be

Chapter H.48

Simon Doclo received the Ph.D. degree in electrical engineering from the Katholieke Universiteit Leuven, Belgium, in 2003. His research interests are in microphone array processing for speech enhancement and source localization, adaptive filtering, computational auditory scene analysis and hearing aid processing. Dr. Doclo received a Best Student Paper Award at the International Workshop on Acoustic Echo and Noise Control in 2001, and the EURASIP Signal Processing Best Paper Award in 2003.

Jasha Droppo

Microsoft Research
Speech Technology Group
Redmond, WA, USA
jdroppo@microsoft.com

Chapter E.33

Jasha Droppo received his Ph.D. from the University of Washington, Seattle, in 2000. His thesis developed a discrete theory of time-frequency distributions and non-stationary signal classification, with an application to speech recognition. Since graduation, he has worked at Microsoft Research, Redmond, on noise robust speech recognition, robust audio capture, speech enhancement, and acoustic modelling for automatic speech recognition.

Thierry Dutoit

Faculté Polytechnique de Mons FPMs
TCTS Laboratory
Mons, Belgium
thierry.dutoit@fpms.ac.be

Chapter D.21

Thierry Dutoit graduated as an electrical engineer and earned his Ph.D. in 1988 and 1993 from the Faculté Polytechnique de Mons, Belgium, respectively, where he is now a full professor. He spent 16 months as a consultant for AT&T Labs Research in Murray Hill and Florham Park, NJ, from July, 1996 to September, 1998. He is the author of two books on speech processing and text-to-speech synthesis, and the coordinator of the MBROLA project for free multilingual speech synthesis.

Gary W. Elko

mh acoustics LLC
Summit, NJ, USA
gwe@mhacoustics.com

Chapter I.50

Gary W. Elko studied electrical engineering at Cornell University. He continued his studies in the area of acoustics and signal processing earning Masters and Doctoral degrees at Penn State University. After graduation, he was employed in the Acoustics Research Department at Bell Labs for more than 15 years. In 2002 he cofounded mh acoustics LLC along with three colleagues.

Sadaoki Furui

Tokyo Institute of Technology Street
Department of Computer Science
Tokyo, Japan
furui@cs.titech.ac.jp

Chapter E.32

Dr. Sadaoki Furui is engaged in a wide range of research on speech analysis, speech recognition, speaker recognition, speech synthesis, and multimodal human – computer interaction and has authored or co-authored over 800 published articles. He has received Paper Awards and Achievement Awards from the IEEE, the IEICE, the ASJ, the Minister of Science and Technology, and the Minister of Education, and the Purple Ribbon Medal from the Japanese Emperor.

Sharon Gannot

Bar-Ilan University
School of Electrical Engineering
Ramat-Gan, Israel
gannot@eng.biu.ac.il

Chapters B.8, H.44, H.47

Sharon Gannot received his B.Sc. from the Technion, Israel in 1986 and the M.Sc. and Ph.D. from Tel-Aviv University, Israel, in 1995 and 2000, respectively, all in Electrical Engineering. In 2001 he held a post-doctoral position at K.U. Leuven, Belgium. From 2002 to 2003 he was a research fellow at the Technion, Israel. Currently, he is a lecturer in the School of Engineering, Bar-Ilan University, Israel. His research interests include statistical signal processing and speech processing using either single- or multi-microphone arrays.

Mazin E. Gilbert

AT&T Labs, Inc., Research
Florham Park, NJ, USA
mazin@research.att.com

Chapter E.34

Mazin Gilbert is a world expert in the area of spoken language technologies. He has over 18 years of experience in industrial research at Bell Labs and AT&T Labs and in academia at Rutgers University, Liverpool University and Princeton University. He is currently Executive Director at AT&T Labs-Research. He is the author of a book entitled, Artificial Neural Networks for Speech Analysis/Synthesis. He holds 16 US patents, published over 90 papers, and is a recipient of several national and international awards. He is currently pursuing an MBA for Executives at Wharton Business School.

Michael M. Goodwin

Creative Advanced Technology Center
Audio Research
Scotts Valley, CA, USA
mgoodwin@atc.creative.com

Chapter B.12

Michael Goodwin received the B.S. and M.S. degrees in Electrical Engineering and Computer Science (EECS) from the Massachusetts Institute of Technology and the Ph.D. in EECS from the University of California, Berkeley, and is now with the Audio Research Department of the Creative Advanced Technology Center in Scotts Valley, California. His research interests include signal modeling and enhancement, audio coding, spatial audio, and array processing, and he is currently chair of the IEEE Signal Processing Society's Technical Committee on Audio and Electroacoustics.

Volodya Grancharov

Multimedia Technologies
Ericsson Research, Ericsson AB
Stockholm, Sweden
volodya.grancharov@ericsson.com

Chapter A.5

Dr. Grancharov is a research engineer in the Multimedia Technologies, Ericsson Research, Stockholm, Sweden. He holds a Ph.D. in telecommunication from the Sound and Image Processing (SIP) Laboratory at the Royal Institute of Technology (KTH), Stockholm, Sweden. His research interests are in the fields of audio and video compression and quality assessment.

Björn Granström

Royal Institute of Technology (KTH)
Department for Speech, Music and Hearing
Stockholm, Sweden
bjorn@speech.kth.se

Chapter D.20

Björn Granström is Professor in Speech Communication at the Royal Institute of Technology (KTH) since 1987. Early research together with Rolf Carlson resulted in a multi-lingual text-to-speech system in 1976. This formed the base of the speech technology company Infovox in 1982. He has published numerous papers in the speech research and technology area, including work in audio-visual speech technology.

Patrick Haffner

AT&T Labs–Research
IP and Voice Services
Middletown, NJ, USA
haffner@research.att.com

Chapter E.29

Patrick Haffner is Lead Member of Technical Staff at AT&T Labs-Research. With Yann LeCun and Vladimir Vapnik, he pioneered the use of Machine Learning (including SVMs and FSMs) for document and image processing applications and is one of the inventors of the DjVu platform. He currently works on algorithms and software solutions for speech, language and sequence processing, with a particular focus on global and large-scale solutions.

Roar Hagen

Global IP Solutions
Stockholm, Sweden
roar.hagen@gipscorp.com

Chapter C.15

Dr. Roar Hagen is co-founder and Chief Technology Officer for Global IP Solutions, a company developing media processing software for IP Communication. Roar began R&D within speech processing and coding in 1989. He previously held positions at AT&T Bell Labs and Ericsson Research. He holds a doctorate in Electrical Engineering from Chalmers University of Technology, Sweden, and an M.Sc. in Physics from the Norwegian Institute of Technology, Norway. Roar Hagen has filed more than 10 patents.

Mary P. Harper

University of Maryland
Center for Advanced Study of Language
College Park, MD, USA
mharper@casl.umd.edu

Chapter G.40

Dr. Mary Harper obtained her Ph.D. in Computer Science from Brown University in 1990. She currently holds the rank of Professor in the School of Electrical and Computer Engineering at Purdue University and is a Senior Research Scientist at the Center for Advanced Study of Language at the University of Maryland. Dr. Harper's research focuses on computer modeling of human communication with a focus on methods for incorporating multiple types of knowledge sources, including lexical, syntactic, prosodic, and most recently visual sources.

Jürgen Herre

Fraunhofer Institute for Integrated Circuits (Fraunhofer IIS)
Audio and Multimedia
Erlangen, Germany
hrr@iis.fraunhofer.de

Chapter C.18

Dr. Jürgen Herre is the Chief Scientist for the Audio/Multimedia activities at Fraunhofer IIS, Erlangen, Germany. He has been working on the development of advanced multimedia technologies and, especially, perceptual audio coding algorithms since 1989 and contributed to many well-known audio coding algorithms, including MPEG-1 Layer 3 (MP3), Advanced Audio Coding (AAC), MPEG-4 Audio and MPEG Surround.

Wolfgang J. Hess

University of Bonn
Institute for Communication Sciences,
Dept. of Communication, Language,
and Speech
Bonn, Germany
wgh@ifk.uni-bonn.de

Chapter B.10

Wolfgang Hess has been a professor of phonetics and speech communication at the University of Bonn since 1986. He received his Dr.-Ing. degree in Electrical Engineering in 1972 and the venia legendi for digital signal processing in 1980, both from the Technical University of Munich. His research areas are speech analysis, especially pitch determination, speech synthesis, and phonetics.

Kiyoshi Honda

Université de la Sorbonne
Nouvelle-Paris III
Laboratoire de Phonétique
et de Phonologie,
ATR Cognitive Information Laboratories
Paris, France
honda@atr.jp

Chapter A.2

Kiyoshi Honda is a specialist in speech production with medical background and research career at the University of Tokyo, Haskins Labs, University of Wisconsin, and ATR. His main interest is the discovery of speech production mechanisms using medical techniques such as EMG and MRI. His major works include mechanisms of vocal control and vowel articulation. Since recently he works on vocal-tract modelling and phonetic instrumentation techniques.

Yiteng (Arden) Huang

Bell Laboratories
Alcatel-Lucent
Murray Hill, NJ, USA
arden_huang@ieee.org

Chapters 1, B.6, B.7, B.13, H.43, H.46, I.51

Dr. Yiteng (Arden) Huang received his Ph.D. in Electrical and Computer Engineering from Georgia Tech in 2001. Upon graduation, he joined Bell Laboratories, and has been, so far, working there as a member of Technical Staff. He has extensively published in acoustic MIMO signal processing for speech and has authored/edited 3 books. He received the 2002 Young Author Best Paper Award from the IEEE Signal Processing Society, has served on several editorial boards/technical committees, and has helped organize a couple of international IEEE workshops.

Matthieu Hébert

Network ASR Core Technology
Nuance Communications
Montréal, Québec, Canada
hebert@nuance.com

Chapter F.37

Dr. Hébert received a Ph.D. in theoretical physics from the Université de Sherbrooke in 1995. After a post-doctoral appointment at Collège de France in Professor de Gennes' laboratory, he was hired by Nortel Networks in the Speech Division. In 1999, he joined Nuance Communications' R+D team. In 2002, he assumed the R+D lead for Nuance Verifier. His research interests are in speaker and speech recognition as well as natural language understanding.

Biing-Hwang Juang

Georgia Institute of Technology
School of Electrical
& Computer Engineering
Atlanta, GA, USA
juang@ece.gatech.edu

Chapter E.26

Biing Hwang (Fred) Juang received his Ph.D. from the University of California, Santa Barbara. After over two decades of career at Bell Laboratories, he joined Georgia Institute of Technology in 2001, holding the Motorola Foundation Chair Professorship. Professor Juang has received numerous technical awards, including the Technical Achievement Award from the Signal Processing Society of the IEEE, and the IEEE Third Millennium Medal. He is a Fellow of the IEEE, a Fellow of Bell Laboratories, a member of the National Academy of Engineering and an Academician of Academia Sinica.

Tatsuya Kawahara

Kyoto University
Academic Center
for Computing and Media Studies
Kyoto, Japan
kawahara@i.kyoto-u.ac.jp

Chapter E.32

Tatsuya Kawahara received the B.E. degree in 1987, the M.E. degree in 1989, and the Ph.D. degree in 1995, all in information science from Kyoto University, Japan. Currently, he is a Professor in the Academic Center for Computing and Media Studies and an Adjunct Professor in the School of Informatics, Kyoto University. He has published more than 100 technical papers covering speech recognition, confidence measures, and spoken dialogue systems.

Ulrik Kjems

Oticon A/S
Smørum, Denmark
uk@oticon.dk

Chapter I.52

Ulrik Kjems received his Ph.D. in 1998 and worked two years as a post doc researcher at the Technical University of Denmark, Department of Mathematical Modelling, on topics of statistical analysis of brain scan images. Since 2000 he is a DSP and algorithm designer at hearing aid manufacturer Oticon, working with noise suppression, speech enhancement and beam-forming systems.

Esther Klabbers

Oregon Health & Science University
Center for Spoken Language
Understanding,
OGI School of Science and Engineering
Beaverton, OR, USA
klabbers@cslu.ogi.edu

Chapter D.23

Dr. Esther Klabbers is an expert in the speech synthesis field. She has a Ph.D. from the Eindhoven University of Technology, the Netherlands. Dr. Klabbers currently works as an Assistant Scientist at the Center for Spoken Language Understanding. Her expertise involves many areas of speech synthesis, such as prosody modeling, corpus design, perceptual evaluations and health-related applications of speech synthesis.

W. Bastiaan Kleijn

Royal Institute of Technology (KTH)
School of Electrical Engineering,
Sound and Image Processing Lab
Stockholm, Sweden
bastiaan.kleijn@ee.kth.se

Chapters A.5, C.14, C.15

Bastiaan Kleijn is a Professor at the School of Electrical Engineering at KTH (Royal Institute of Technology) in Stockholm, Sweden and heads the Sound and Image Processing Laboratory. He is also a founder and former Chairman of Global IP Solutions where he remains Chief Scientist. He holds a Ph.D. in Electrical Engineering from Delft University of Technology (Netherlands), a Ph.D. in Soil Science and an M.S. in Physics, both from the University of California, and an M.S. in Electrical Engineering from Stanford University. He worked on speech processing at AT&T Bell Laboratories from 1984 to 1996, first in development and later in research. He is a Fellow of the IEEE.

Birger Kollmeier

Universität Oldenburg
Medizinische Physik
Oldenburg, Germany
birger.kollmeier@uni-oldenburg.de

Chapter A.4

Birger Kollmeier obtained his Ph.D. in physics and his Ph.D. in medicine in Göttingen, Germany. Since 1993 he is a full professor in physics and head of the medical physics group at the Universität Oldenburg performing fundamental and applied research in psychoacoustics, speech processing, hearing aids and auditory neuroscience. He is the chairman of the translational research institutions Hörzentrum Oldenburg and HörTech, i.e. national center of excellence in hearing technology and audioloy, and was awarded several prizes.

Ermin Kozica

Royal Institute of Technology (KTH)
School of Electrical Engineering,
Sound and Image Processing Laboratory
Stockholm, Sweden
ermin.kozica@ee.kth.se

Chapter C.15

Ermin Kozica received his M.Sc. degree in Electrical Engineering from KTH in January 2006. Since then, he is a graduate student in the Sound and Image Processing Lab., School of Electrical Engineering at KTH. The working title of his thesis is *Methods for Robust Video Coding*. His research interests include video processing, image processing and signal processing in general.

Sen M. Kuo

Northern Illinois University
Department of Electrical Engineering
DeKalb, IL, USA
kuo@ceet.niu.edu

Chapter H.49

Dr. Kuo is a Professor and Chair of Electrical Engineering at the Northern Illinois University, DeKalb, IL. He received the B.S. degree from the National Taiwan Normal University, Taipei, Taiwan, in 1976 and the M.S. and Ph.D. degrees from the University of New Mexico in 1983 and 1985, respectively. In 1993, he was with Texas Instruments, Houston, TX.He serves as an associate editor for IEEE Transactions on Audio, Speech and Language Processing since June 2006 to present. Dr. Kuo is the leading author of five books, has been awarded seven US patents and has published over 190 technical papers. His research focuses on active noise and vibration control, real-time DSP systems, adaptive echo and noise cancellation, biomedical signal processing, and digital communication and audio applications.

Jan Larsen

Technical University of Denmark
Informatics and Mathematical Modelling
Kongens Lyngby, Denmark
jl@imm.dtu.dk

Chapter I.52

Dr. Jan Larsen received Ph.D. degree from the Technical University of Denmark (DTU) in 1994 and is currently Associate Professor at Informatics and Mathematical Modelling, DTU. He has authored and co-authored more than 100 papers and book chapters in the areas of machine learning for signal processing, and has been involved in many IEEE conference organizations.

Chin-Hui Lee

Georgia Institute of Technology
School of Electrical
and Computer Engineering
Atlanta, GA, USA
chl@ece.gatech.edu

Chapters G.39, G.42

Dr. Chin-Hui Lee received his Ph.D. degree from University of Washington in 1981. His current research interests include multimedia communication, speech and language processing, biometric authentication, and information retrieval. In 2007 he received a Technical Achievement Award from the IEEE Signal Processing Society for "Exceptional Contributions to the Field of Automatic Speech Recognition". Dr. Lee is a fellow of the IEEE.

Haizhou Li

Institute for Infocomm Research
Department of Human Language
Technology
Singapore
hli@i2r.a-star.edu.sg

Chapter G.42

Dr. Haizhou Li received his Ph.D degree from the South China University of Technology in 1990. He is now the Head of Human Language Technology Department at the Institute for Infocomm Research, Singapore. His research interests include automatic speech recognition, speaker and language recognition, and natural language processing. Dr Li is a recipient of the National Infocomm Award 2001 in Singapore. He is the Vice President of COLIPS and a Senior Member of IEEE.

Jan Linden

Global IP Solutions
San Francisco, CA, USA
jan.linden@gipscorp.com

Chapter C.15

Jan Linden is the Vice President of Engineering at Global IP Solutions. He has been conducting research and development in speech processing and communications for more than 15 years. Prior to joining Global IP Solutions he was with the University of California, Santa Barbara and SignalCom. He holds a Ph.D. and an M.Sc. in Electrical Engineering from Chalmers University of Technology, Sweden.

Manfred Lutzky

Fraunhofer Integrated Circuits (IIS)
Multimedia Realtime Systems
Erlangen, Germany
manfred.lutzky@iis.fraunhofer.de

Chapter C.18

Manfred Lutzky received his M.S. (Diplom) degree in Electrical Engineering from Erlangen University and joined the Fraunhofer Institute for Integrated Circuits (IIS) in Erlangen, Germany, in 1997. He was engaged in the implementation of the audio coding schemes mp3 and MPEG-4 AAC on embedded platforms. Since 2001 he is in the position of a group manager now heading the Audio for Communcation group that deals with research, implementation and optimization of low-delay audio codecs such as MPEG-4 AAC-LD and ULD and related technology for communication applications. He is actively participating in the standardization of audio codecs in ETSI/NGDect body.

Bin Ma

Human Language Technology
Institute for Infocomm Research
Singapore
mabin@i2r.a-star.edu.sg

Chapter G.42

Dr. Ma is a Research Scientist in the Speech and Dialogue Processing group, Institute for Infocomm Research (I^2R), Singapore. Prior to joining I^2R, he worked in speech recognition at the Chinese Academy of Sciences, Lernout & Hauspie Speech Products, and InfoTalk Corporation. He received his Ph.D. degree from The University of Hong Kong in speech recognition. He is a Senior Member of IEEE.

Michael Maxwell

University of Maryland
Center for Advanced Study of Language
College Park, MD, USA
mmaxwell@casl.umd.edu

Chapter G.40

Dr. Maxwell is a Senior Research Scientist at the Center for Advanced Study of Language at the University of Maryland. He obtained his Ph. D. in Linguistics from the University of Washington in 1984. Prior to joining the University of Maryland, his research included describing endangered languages of Ecuador and Colombia (under the auspices of the Summer Institute of Linguistics), computational syntax, morphology and phonology, and the building of resources for low-density languages (at the Linguistic Data Consortium of the University of Pennsylvania).

Alan V. McCree

MIT Lincoln Laboratory
Department of Information Systems Technology
Lexington, MA, USA
mccree@ll.mit.edu

Chapter C.16

Dr. Alan McCree has been with MIT Lincoln Laboratory since 2004, where he conducts research in speech coding, noise suppression, and speaker recognition. He has also worked at Texas Instruments, AT&T Bell Laboratories, and M/A-COM Linkabit. He is an IEEE Fellow. Dr. McCree has published 40 technical papers and has been awarded 30 U. S. patents.

Bernd Meyer

Carl von Ossietzky Universität Oldenburg
Medical Physics Section, Haus des Hörens
Oldenburg, Germany
bernd.meyer@uni-oldenburg.de

Chapter A.4

Bernd Meyer is a Ph.D. student at the Medical Physics Section at the University of Oldenburg. His research activities focus on the speech recognition performance of human listeners compared to machines and the improvement of feature extraction of automatic speech recognition systems.

Jens Meyer

mh acoustcis
Summit, NJ, USA
jmm@mhacoustics.com

Chapter I.50

Jens Meyer studied electrical engineering at Darmstadt University of Technology. He continued there with his Dr.-Ing where he specialized in acoustic signal processing. He stayed in this field ever since, focusing on multi-microphone application for communication and sound recording. In 2002 together with 3 colleagues, he co-founded mh acoustics.

Taniya Mishra

Oregon Health and Science University
Center for Spoken Language
Understanding, Computer Science
and Electrical Engineering,
OGI School of Science and Engineering
Beaverton, OR, USA
mishra@cslu.ogi.edu

Chapter D.23

Taniya Mishra is a doctoral student at the Center for Spoken Language Understanding at the OGI School of Science &Engineering at Oregon Health and Science University. Her main research focus is intonation modelling and speech synthesis. For her dissertation, she is working on developing an algorithm for decomposing pitch contours into their component curves, using the general assumptions of the superpositional model of intonation.

Mehryar Mohri

Courant Institute of Mathematical
Sciences
New York, NY, USA
mohri@cs.nyu.edu

Chapters E.28, E.29

Mehryar Mohri is a Professor of Computer Science at the Courant Institute of Mathematical Sciences and a Research Consultant at Google Research. Prior to these positions, he spent about ten years at AT&T Bell Labs or AT&T Labs - Research (1995-2004) where, in the last four years, he served as the Head of the Speech Algorithms Department and as a Technology Leader, overseeing research projects in machine learning, text and speech processing, and the design of general algorithms. Mehryar Mohri graduated from Ecole Polytechnique in France in 1987, received his M.S. degree in Mathematics and Computer Science from Ecole Normale Superieure d'Ulm (1988), and his M.S. degree (1989) and Ph.D. (1993) in Computer Science from the University of Paris 7. His current topics of interest are machine learning, computational biology, and text and speech processing.

Marc Moonen

Katholieke Universiteit Leuven
Electrical Engineering Department ESAT/
SISTA
Leuven, Belgium
marc.moonen@esat.kuleuven.be

Chapter H.48

Marc Moonen is a Professor of Signal Processing at the Electrical Engineering Department of Katholieke Universiteit Leuven He is a Fellow of the IEEE. He is currently President of the European Association for Signal Processing (EURASIP). He has served as Editor-in-Chief for the EURASIP Journal on Applied Signal Processing (2003-2005), and is/has been a member of the editorial board of five other journals.

Dennis R. Morgan

Bell Laboratories, Alcatel-Lucent
Murray Hill, NJ, USA
drrm@bell-labs.com

Chapter H.49

Dr. Morgan received the B.S. degree in 1965 from the University of Cincinnati, OH, and the M.S. and Ph.D. degrees from Syracuse University, Syracuse, NY, in 1968 and 1970, respectively, all in electrical engineering. He is a Distinguished Member of Technical Staff with Bell Laboratories, Alcatel-Lucent where he has been employed since 1984, and is involved in research on adaptive signal processing applied to RF and optical communication systems. Before that, he was with General Electric Company, Electronics Laboratory, Syracuse, NY, specializing in the analysis and design of signal processing systems used in radar, sonar, and communications. He has authored numerous journal publications and is coauthor of two books.

David Nahamoo

IBM Thomas J. Watson Research Center
Yorktown Heights, NY, USA
nahamoo@us.ibm.com

Chapter E.30

David Nahamoo is the Chief Technical Officer and Business Strategist for Speech at the IBM Thomas J. Watson Research Center. He has worked in the Speech Recognition area since 1982, joining IBM after finishing his doctorate at Purdue. He is a Fellow of the IEEE and has received two best paper awards from the IEEE Signal Processing Society.

Douglas O'Shaughnessy

Université du Québec
INRS Énergie, Matériaux
et Télécommunications (INRS-EMT)
Montréal, Québec, Canada
dougo@emt.inrs.ca

Chapter B.11

Douglas O'Shaughnessy is Professor at INRS-EMT and adjunct Professor at McGill University. His interests include automatic speech synthesis, analysis, coding, enhancement, and recognition. A graduate of MIT, he is a Fellow of the IEEE and the Acoustical Society of America (ASA). He is a member of the IEEE Technical Comittee for Speech Processing, and was the General Chair of the 2004 International Conference on Acoustics, Speech and Signal Processing. He is the author of *Speech Communications: Human and Machine.*

Lucas C. Parra

Steinman Hall,
The City College of New York
Department of Biomedical Engineering
New York, NY, USA
parra@ccny.cuny.edu

Chapter I.52

Dr. Parra is Associate Professor of Biomedical Engineering at The City College of New York (CCNY). Previously he was at Sarnoff Corporation and at Siemens Corporate Research. He is a physicist with expertise in acoustic array processing, emission tomography, electroencephalography, machine learning and pattern recognition. His current research focuses on functional brain imaging and computational models of the central nervous system.

Sarangarajan Parthasarathy

Yahoo!, Applied Research
Sunnyvale, CA, USA
parthas@yahoo-inc.com

Chapter F.36

Dr. Sarangarajan Parthasarathy has spent much of his research career at AT&T Bell Labs and its successors, contributing to advances in many areas of speech processing including articulatory modelling, automatic speech and speaker recognition, and multimedia processing. His responsibilities ranged from basic research to the implementation of algorithms and system prototypes. He was a member of the Speech Technical Committee of the IEEE Signal Processing Society from 2002 to 2006. He recently joined the applied research group at Yahoo to work on statistical modelling applied to web/sponsored search.

Michael Syskind Pedersen

Oticon A/S
Smørum, Denmark
msp@oticon.dk

Chapter I.52

Michael Syskind Pedersen obtained his Ph.D. in 2006 from the department of Informatics and Mathematical Modelling (IMM) at the Technical University of Denmark (DTU). In 2005 he was a visiting scholar at the department of Computer Science and Engineering at The Ohio State University, Columbus OH. His main areas of research are multi-microphone audio processing and noise reduction. Since 2001 Michael has been employed with the hearing aid company Oticon.

Fernando Pereira

University of Pennsylvania
Department of Computer
and Information Science
Philadelphia, PA, USA
pereira@cis.upenn.edu

Chapter E.28

Fernando Pereira is the Andrew and Debra Rachleff Professor and chair of the department of Computer and Information Science, University of Pennsylvania. He received a Ph.D. in Artificial Intelligence from the University of Edinburgh in 1982. Before joining Penn, he held industrial research and management positions at SRI International, at AT&T Labs, and at WhizBang Labs, a Web information extraction company. His main research interests are in machine-learnable models of language and biological sequences. He has over 100 research publications and several patents. He was elected Fellow of the American Association for Artificial Intelligence in 1991 for his contributions to computational linguistics and logic programming, and he was president of the Association for Computational Linguistics.

Michael Picheny

IBM Thomas J. Watson Research Center
Yorktown Heights, NY, USA
Picheny@us.ibm.com

Chapter E.30

Michael Picheny is the Senior Manager of the Speech and Language Algorithms Group at the IBM Thomas J. Watson Research Center. He has worked in the Speech Recognition area since 1981, joining IBM after finishing his doctorate at MIT. He is a Fellow of the IEEE and a member of the board of ISCA.

Rudolf Rabenstein

University Erlangen-Nuremberg
Electrical Engineering, Electronics,
and Information Technology
Erlangen, Germany
rabe@LNT.de

Chapter I.53

Rudolf Rabenstein studied electrical engineering in Erlangen, Germany and Boulder, Colorado, USA. He worked with the Physics Department of the University of Siegen, Germany and currently with the Telecommunications Laboratory of the University Erlangen-Nuremberg, Germany. Here he obtained his doctoral degree and the "Habilitation" degree in signal processing. His research interests are in the fields of multidimensional systems theory and multimedia signal processing.

Lawrence Rabiner

Rutgers University
Department of Electrical
and Computer Engineering
Piscataway, NJ, USA
lrr@caip.rutgers.edu

Chapter E.26

Lawrence Rabiner received the S.B., and S.M. degrees simultaneously in June 1964, and the Ph.D. degree in Electrical Engineering in June 1967, all from MIT. Dr. Rabiner joined AT&T Bell Labs in 1967 as a Member of the Technical Staff and served in various positions. In 1996 he joined AT&T Labs in 1996 as Director of the Speech and Image Processing Services Research Lab, and was promoted to Vice President of Research in 1998. Dr. Rabiner retired at the end of March 2002 and is now a Professor of Electrical and Computer Engineering at Rutgers University, and the Associate Director of the Center for Advanced Information Processing (CAIP). He also has a joint appointment as a Professor of Electrical and Computer Engineering at the University of California at Santa Barbara.

Douglas A. Reynolds

Massachusetts Institute of Technology
Lincoln Laboratory,
Information Systems Technology Group
Lexington, MA, USA
dar@ll.mit.edu

Chapters F.38, G.41

Dr. Douglas Reynolds is a Senior Member of Technical Staff at MIT Lincoln Laboratory with over 20 years experience in research, development, evaluation and deployment of speaker, language and speech recognition technologies for commercial and governmental applications. He invented several widely used algorithms in robust speaker and language recognition and pioneered approaches to exploit high-level information for improved speaker recognition accuracy. His current research efforts are focused on robust, portable speaker and language recognition and synthesis of paralinguistic information for characterizing large audio data sets.

Michael Riley

Google, Inc., Research
New York, NY, USA
riley@google.com

Chapter E.28

Michael Riley received his Ph.D. in computer science from MIT in 1987. He joined Bell Labs in Murray Hill, NJ in 1987 and moved to AT&T Labs in Florham Park, NJ in 1996. He is currently a member of the research staff at Google, Inc. in New York City. His interests include speech and natural language processing, text analysis, information retrieval, and machine learning.

Aaron E. Rosenberg

Rutgers University
Center for Advanced Information Processing
Piscataway, NJ, USA
aer@caip.rutgers.edu

Chapter F.36

Aaron Rosenberg is a Research Professor at the Center for Advanced Information Processing (CAIP) at Rutgers University. He retired from AT&T Labs Research as a Technology Leader in the Speech and Image Processing Services Laboratory. Prior to his service at AT&T, he was a Distinguished Member of the Technical Staff at Bell Laboratories. His research activities have included auditory psychophysics, speech perception, speech quality, and speech and speaker recognition. He has authored or co-authored some 100 papers and been granted 12 patents in these fields. He is a Fellow of the Acoustical Society of America and the IEEE.

Salim Roukos

IBM T. J. Watson Research Center
Multilingual NLP Technologies
Yorktown Heights, NY, USA
roukos@us.ibm.com

Chapter E.31

Salim Roukos is Senior Manager and CTO for Translation Technologies at IBM. His research areas at IBM have been in statistical machine translation, information extraction, statistical parsing, and statistical language understanding for conversational systems. Roukos received his B.E. from the American University of Beirut, in 1976, his M.Sc. and Ph. D. from the University of Florida, in 1978 and 1980, respectively. Roukos has served as Chair of the IEEE Digital Signal Processing Committee in 1988. Roukos lead the group that created IBM's ViaVoice Telephony product, the first commercial software to support full natural language understanding for dialog systems in 2000, and more recently the first statistical machine translation product for Arabic-English translation in 2003.

Jan van Santen

Oregon Health And Science University
OGI School of Science and Engineering,
Department of Computer Science
and Electrical Engineering
Beaverton, OR, USA
vansanten@cslu.ogi.edu

Chapter D.23

Dr. van Santen's current responsibilities consist of teaching, conducting research, and managing the Center for Spoken Language Understanding (CSLU) as well as the Computer Science and Electrical Engineering (CSEE) Department, both at the OGI School of Science and Engineering at the Oregon Health and Science University in Portland, Oregon. His background includes mathematical modeling of prosody, signal processing, and computational linguistics. He has seven patents and over 100 publications. Dr. van Santen is currently the Director of CSLU, the Chair of the CSEE Department, serves on the board of directors of SVOX AG, and is president of BioSpeech Inc. He has a Ph.D. degree in Mathematical Psychology from the University of Michigan.

Ronald W. Schafer

Hewlett-Packard Laboratories
Palo Alto, CA, USA
ron.schafer@hp.com

Chapter B.9

Ronald W. Schafer is Professor Emeritus at Georgia Tech and a HP Fellow in the Mobile and Media Systems Laboratory at Hewlett-Packard Laboratories in Palo Alto, CA. He has co-authored six widely used textbooks in the DSP field, and has received numerous awards for teaching and research including the IEEE Education Medal and membership in the National Academy of Engineering.

Juergen Schroeter

AT&T Labs – Research
Department of Speech Algorithms and Engines
Florham Park, NJ, USA
schroeter@att.com

Chapter D.19

Juergen Schroeter earned his Dr.-Ing. in Electrical Engineering from Ruhr-University in Bochum, Germany, in 1983. As a Member of Technical Staff at AT&T Bell Laboratories in Murray Hill, NJ, he worked on speech coding and speech synthesis methods that employ computational models of the vocal tract and glottis. At AT&T Labs – Research in Florham Park, NJ, he is responsible for research in speech algorithms and engines. His group created the AT&T Natural Voices® Text-to-Speech system in 2001. In the same year, he received the AT&T Science and Technology Medal. He is a Fellow of the Acoustical Society of America and a Fellow of the IEEE.

Stephanie Seneff

Massachusetts Institute of Technology
Computer Science and Artificial
Intelligence Laboratory
Cambridge, MA, USA
senef@csail.mit.edu

Chapter E.35

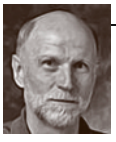
Dr. Stephanie Seneff's research interests encompass a broad range of topics centered around spoken conversational interfaces to computers, including speech recognition, natural language understanding, discourse and dialogue modelling, natural language generation, and language translation. She has served as a member of the Permanent Council for the International Conference on Spoken Language Systems (ICSLP).

Wade Shen

Massachusetts Institute of Technology
Communication Systems, Information
Systems Technology, Lincoln Laboratory
Lexington, MA, USA
swade@ll.mit.edu

Chapter G.41

Wade Shen received his Master's degree in Computer Science from the University of Maryland, College Park in 1997, and his Bachelor's degree in Electrical Engineering and Computer Science from the University of California, Berkeley in 1994. His current areas of research involve machine translation and machine translation evaluation, speech, speaker, and language recognition for small-scale and embedded applications, named-entity extraction, and prosodic modeling. Prior to joining Lincoln Laboratory in 2003, Wade helped found and served as Chief Techology Officer for Vocentric Corporation, a company specializing in speech technologies for small devices.



Elliot Singer

Massachusetts Institute of Technology
Information Systems Technology Group,
Lincoln Laboratory
Lexington, MA, USA
es@ll.mit.edu

Chapter G.41

Elliot Singer received an S.M. degree in electrical engineering from MIT in 1974 and is currently a staff member of the MIT Lincoln Laboratory Information Systems Technology Group. Since 1977 he has worked in a wide variety of speech related areas, including low rate speech coding, custom hardware development, keyword spotting, pattern recognition, and speaker and language recognition.

Jan Skoglund

Global IP Solutions
San Francisco, CA, USA
jan.skoglund@gipscorp.com

Chapter C.15

Jan Skoglund received his Ph.D. degree from Chalmers University of Technology, Sweden. From 1999 to 2000, he worked on low bit rate speech coding as a consultant at AT&T Labs-Research, Florham Park, NJ. Since 2000, he has been with Global IP Solutions, San Francisco, CA, where he is working on speech and audio processing tailored for packet-switched networks.

M. Mohan Sondhi

Avayalabs Research
Basking Ridge, NJ, USA
mms@research.avayalabs.com

Chapters 1, H.45

M. Mohan Sondhi is a consultant at Avaya Research Labs, Basking Ridge, NJ. Prior to joining Avaya he spent 39 years at Bell Labs, from where he retired in 2001. He holds undergraduate degrees in Physics and Electrical Communication Engineering, and M.S and Ph.D. degrees in Electrical Engineering. At Bell Labs he conducted research in speech signal processing, echo cancellation, acoustical inverse problems, speech recognition, articulatory models for analysis and synthesis of speech and modeling of auditory and visual processing by humans.

Sascha Spors

Deutsche Telekom AG, Laboratories
Berlin, Germany
Sascha.Spors@telekom.de

Chapter I.53

Dr. Sascha Spors is senior research scientist at the Deutsche Telekom Laboratories. He obtained his doctoral degree in Electrical Engineering from the University Erlangen-Nuremberg in January 2006. His research background is on wavefield analysis, multichannel sound reproduction and efficient algorithms for massive multichannel adaptation problems. His special interest is on wave field synthesis, where he has implemented several laboratory systems.

Ann Spriet

ESAT-SCD/SISTA, K.U. Leuven
Department of Electrical Engineering
Leuven, Belgium
ann.spriet@esat.kuleuven.be

Chapter H.48

Dr. A. Spriet is a Postdoctoral Fellow of the Fund for Scientific Research-Flanders, affiliated with the Electrical Engineering Department (ESAT) and the Department of Neurosciences (ExpORL) of the Katholieke Universiteit Leuven. Her research interests are in digital signal processing for speech enhancement in hearing aids and cochlear implants. She received the 2001 Lapperre Award and a Best Student Paper Award at IWAENC in 2003 and ICASSP in 2005.

Richard Sproat

University of Illinois
at Urbana-Champaign
Department of Linguistics
Urbana, IL, USA
rws@uiuc.edu

Chapter D.22

Richard Sproat received his Ph.D. in Linguistics from the Massachusetts Institute of Technology in 1985. Since then he has worked at AT&T Bell Labs, at Lucent's Bell Labs and at AT&T Labs – Research, before joining the faculty of the University of Illinois. Sproat has worked in numerous areas relating to language and computational linguistics, including syntax, morphology, computational morphology, articulatory and acoustic phonetics, text processing, text-to-speech synthesis, writing systems, and text-to-scene conversion, and has published widely in these areas. His most recent work includes work on the prediction of emotion from text for TTS, multilingual named entity transliteration, and the effects of script layout on readers' phonological awareness.

Yannis Stylianou

Institute of Computer Science
Heraklion, Crete, Greece
yannis@csd.uoc.gr

Chapter D.24

Yannis Stylianou is Associate Professor at University of Crete, Department of Computer Science. He received the Diploma of Electrical Engineering from the National Technical University, NTUA, of Athens in 1991 and the M.Sc. and Ph.D. degrees in Signal Processing from the Ecole National Superieure des Telecommunications, ENST, Paris, France in 1992 and 1996, respectively. From 1996 until 2001 he was with AT&T Labs Research (Murray Hill and Florham Park, NJ, USA) as a Senior Technical Staff Member. In 2001 he joined Bell-Labs Lucent Technologies, in Murray Hill, NJ, USA. Since 2002 he is with the Computer Science Department at the University of Crete.

Jes Thyssen

Broadcom Corporation
Irvine, CA, USA
jthyssen@broadcom.com

Chapter C.17

Jes Thyssen received his Ph.D. from the Technical University of Denmark. He is a Senior Principle Scientist with Broadcom working in the area of speech and audio processing. He holds numerous patents in the field and has contributed to a number of speech coding standards. Current research interests include packetloss concealment, noise suppression, blind source separation, and speech coding.

Jay Wilpon

Research AT&T Labs
Voice and IP Services
Florham Park, NJ, USA
jgw@research.att.com

Chapter E.34

Jay Wilpon is Executive Director of Speech Research at AT&T Labs. He has authored over 90 publications. His research led to the first nationwide deployment of both wordspotting and spoken language understanding technologies. Jay was named an IEEE Fellow for his leadership in the development of automatic speech recognition algorithms. He served as Member at Large to the IEEE Signal Processing Society (SPS) Board of Governors, and as Chair of the SPS Speech Processing Technical Committee. For pioneering leadership in the creation and deployment of speech recognition-based services in the telephone network, he was awarded the distinguished honor of AT&T Fellow.

Jan Wouters

ExpORL, Department of Neurosciences,
K.U. Leuven
Leuven, Belgium
jan.wouters@med.kuleuven.be

Chapter H.48

Jan Wouters is a professor at the Department of Neurosciences of the Katholieke Universiteit Leuven, Belgium. He obtained his PhD in Physics in 1989. His research activities are mainly focused on audiology and the signal processing in the auditory system, signal processing for cochlear implants and hearing aids, and electrophysiological and psychophysical measures for assessment. He is author of about 120 articles in peer-review international journals.

Arie Yeredor

Tel-Aviv University
Electrical Engineering - Systems
Tel-Aviv, Israel
arie@eng.tau.ac.il

Chapter B.8

Arie Yeredor received his Ph.D. from Tel-Aviv University (TAU) in 1997. He is currently a Senior Lecturer at TAU, teaching courses in statistical and digital signal processing, and serving as Academic Director of the DSP Labs. His research interests include estimation theory, statistical signal processing and blind source separation.

Steve Young

Cambridge University Engineering Dept
Cambridge, UK
sjy@eng.cam.ac.uk

Chapter E.27

Steve Young is Professor of Information Engineering and Head of the Information Engineering Division at Cambridge University, UK. His main research interests lie in the area of spoken language systems including speech recognition, speech synthesis and dialogue management. He is a Fellow of the Royal Academy of Engineering and the Institution of Electrical Engineers (IEE). Professor Young is a member of the British Computer Society (BCS) and a Senior Member of the IEEE. In 2004, he was a recipient of an IEEE Signal Processing Society Technical Achievement Award.

Victor Zue

Massachusetts Institute of Technology
CSAI Laboratory
Cambridge, MA, USA
zue@csail.mit.edu

Chapter E.35

Victor Zue is the Delta Electronics Professor of Electrical EE&CS at MIT and the Director of the Computer Science and Artificial Intelligence Laboratory. Earlier in his career, Victor conducted research in acoustic phonetics and phonology in American English. Subsequently, his research interest shifted to the development of spoken language interfaces to make human – computer interactions easier and more natural.

Detailed Contents

Detailed Cont.

Part D Text-to-Speech Synthesis

Part F Speaker Recognition

36 Overview of Speaker Recognition

37 Text-Dependent Speaker Recognition

Subject Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

Y

Z