PREDICTIVE LINEAR TRANSFORMS FOR NOISE ROBUST SPEECH RECOGNITION

M.J.F. Gales and R.C. van Dalen

Cambridge University Engineering Department
Cambridge CB2 1PZ, UK

ABSTRACT

It is well known that the addition of background noise alters the correlations between the elements of, for example, the MFCC feature vector. However, standard model-based compensation techniques do not modify the feature-space in which the diagonal covariance matrix Gaussian mixture models are estimated. One solution to this problem, which yields good performance, is Joint Uncertainty Decoding (JUD) with full transforms. Unfortunately, this results in a high computational cost during decoding. This paper contrasts two approaches to approximating full JUD while lowering the computational cost. Both use predictive linear transforms to modify the feature-space: adaptation-based linear transforms, where the model parameters are restricted to be the same as the original clean system; and precision matrix modelling approaches, in particular semi-tied covariance matrices. These predictive transforms are estimated using statistics derived from the full JUD transforms rather than noisy data. The schemes are evaluated on AURORA 2 and a noise-corrupted Resource Management task.

Index Terms— Noise robust speech recognition, joint uncertainty decoding, precision matrix modelling.

1. INTRODUCTION

Speech recognition in noise has been an area of active research for many years. Good performance using model-based compensation schemes, such as Parallel Model Combination (PMC) [1] and Vector Taylor Series (VTS) [2], can be obtained. However, these approaches are computationally expensive compared to front-end enhancement based schemes. Furthermore, standard model-based compensation schemes do not modify the feature-space in which the acoustic models are estimated. As diagonal covariance matrix Gaussian Mixture Models (GMMs) are commonly used to model the state output distributions, changes in the correlations between the elements of the feature-vector may be expected to affect the performance of the system[1]. In particular as the signal-to-noise ratio (SNR) decreases, this may be expected to become more important. In contrast, some feature-space enhancement schemes, such as Probabilistic Optimal Filtering (POF) [3] and versions of SPLICE [4], transform the feature-space, but these are explicitly linked to using stereo training data and a fixed set of basis transforms. This transformation of the feature-space may in some conditions allow feature-compensation schemes to out-perform model-based approaches.

Recently there has been interest in examining an elegant compromise between model-based approaches and front-end schemes, *uncertainty decoding*. These approaches propagate a measure of the uncertainty introduced by the background acoustic noise into the recognition process [5, 6]. One attribute of Joint Uncertainty Decoding (JUD) [6] is that full transforms of the feature-space, that model the changes in the correlation as the noise changes, may be estimated. This yields a model-based scheme that allows changes in the correlations to be modelled. However, when full transformations are used the resultant covariance matrices associated with each Gaussian becomes full with the large associated computational cost during recognition. This paper considers how JUD with full transforms can be efficiently approximated.

Two forms of approach to improving the efficiency of full JUD are examined. The first is based on linear adaptation transforms, such as constrained MLLR (CMLLR) and full-covariance transformations (MLLRCov) [7]. These transforms modify the feature-space, and possibly transform the original "clean" model parameters. The second class is based on efficient precision matrix models [8, 9, 10]. Here the covariance matrix of the component of the state GMMs are modified to better model the correlations in the data without significantly increasing the computational cost. Both of these approaches are referred to as *predictive linear transforms* as their parameters are estimated based on statistics "predicted" from JUD rather than noisy data. This is similar to the predictive model-based compensation schemes which use noise estimates and mismatch functions [11].

This paper is organised as follows. The next section briefly describes JUD. This is followed by a description of predictive linear transforms and how they may be efficiently estimated from the JUD parameters. The computational costs associated with estimating and applying these transforms are then described. Finally, the results on the AURORA 2 and noise corrupted Resource Management tasks are given.

2. JOINT UNCERTAINTY DECODING

This section gives an overview of the uncertainty decoding framework used in [5, 6]. The effects of environmental noise can be represented in a dynamic Bayesian network as shown in figure 1. Here, the noise corrupted speech observation $\boldsymbol{y}_t$ at time t is assumed to be

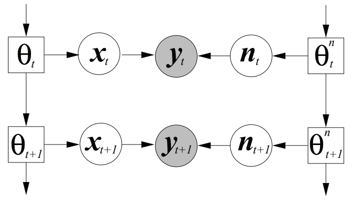

Fig. 1. Uncertainty Decoding Dynamic Bayesian Network.

Thanks to Hank Liao, and funding from Toshiba Research Europe Ltd, for making code available.

[1] PMC and VTS can in theory be used to obtain full covariance matrix systems, but these have the full covariance matrix decoding cost.

978-1-4244-1746-9/07/$25.00 ©2007 IEEE

conditionally independent of all other observations given the clean speech $\boldsymbol{x}_t$ and the noise $\boldsymbol{n}_t$ at that time. The clean speech and noise are assumed to be generated by HMMs with states θ_t^{n} for the noise and θ_t for the clean speech. Under these assumptions the likelihood of the corrupted observation may be expressed as

$$p(\boldsymbol{y}_t|\mathcal{M},\check{\mathcal{M}},\theta_t) = \int p(\boldsymbol{y}_t|\boldsymbol{x}_t,\check{\mathcal{M}})p(\boldsymbol{x}_t|\mathcal{M},\theta_t)d\boldsymbol{x}_t \qquad (1)$$

where

$$p(\boldsymbol{y}_t|\boldsymbol{x}_t,\check{\mathcal{M}}) = \int p(\boldsymbol{y}_t|\boldsymbol{x}_t,\boldsymbol{n}_t)p(\boldsymbol{n}_t|\check{\mathcal{M}},\theta_t^{\mathsf{n}})d\boldsymbol{n}_t \qquad (2)$$

and $\check{\mathcal{M}}$ the front-end compensation model. The acoustic model $\mathcal{M}$ consists of Gaussian components each defined by a prior, c_m, mean, $\boldsymbol{\mu}^{(m)}$, and diagonal covariance matrix, $\boldsymbol{\Sigma}_{\mathsf{diag}}^{(m)}$, so

$$p(\boldsymbol{x}_t|\mathcal{M},\theta_t) = \sum_{m\in\theta_t} c_m\mathcal{N}\left(\boldsymbol{x}_t;\boldsymbol{\mu}^{(m)},\boldsymbol{\Sigma}_{\mathsf{diag}}^{(m)}\right) \qquad (3)$$

The likelihood calculation thus has two distinct parts. Only the first, $p(\boldsymbol{y}_t|\boldsymbol{x}_t,\check{\mathcal{M}})$, is a function of the noise. Equation 1 does not depend on the noise given the form of $p(\boldsymbol{y}_t|\boldsymbol{x}_t,\check{\mathcal{M}})$. Uncertainty decoding takes advantage of this factorisation by using an appropriate form of approximation for the conditional distribution of the corrupted speech given the clean speech for a particular noise environment. As the complexity of this approximation is independent of the complexity of the actual acoustic models, there is a large degree of flexibility in choosing the computational cost of the decoding process.

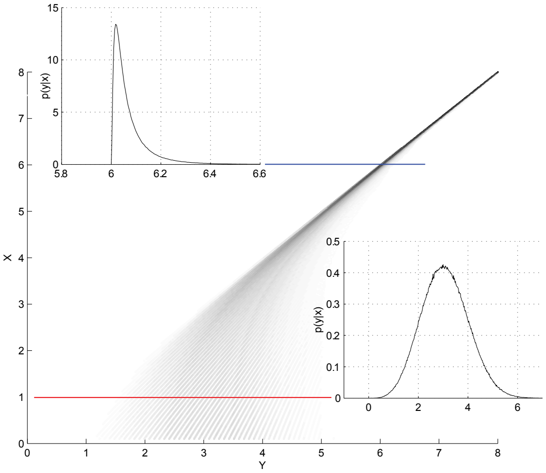

Fig. 2. Joint distribution $p(x, y)$.

For uncertainty decoding schemes, an important consideration is how to partition the feature-space so that simple region-dependent approximations can be used to model the complexity of the joint distribution, $p(\boldsymbol{x}_t,\boldsymbol{y}_t)$. A single dimensional version of this joint distribution is shown in figure 2. Following [12], a model-based joint uncertainty decoding approach is adopted in this work. In this case JUD yields likelihood calculations of the form

$$p(\boldsymbol{y}_t|\mathtt{s}_m) = |\mathbf{A}_{\mathsf{jnt}}^{(r)}|\mathcal{N}\left(\mathbf{A}_{\mathsf{jnt}}^{(r)}\boldsymbol{y}_t + \mathbf{b}_{\mathsf{jnt}}^{(r)};\boldsymbol{\mu}^{(m)},\boldsymbol{\Sigma}_{\mathsf{diag}}^{(m)}+\boldsymbol{\Sigma}_{\mathsf{b}}^{(r)}\right) \qquad (4)$$

where component $\mathtt{s}_m$ "belongs" to base-class $\mathtt{r}_r$,

$$\mathbf{A}_{\mathsf{jnt}}^{(r)} = \boldsymbol{\Sigma}_x^{(r)}\boldsymbol{\Sigma}_{yx}^{(r)\text{-}1}, \qquad (5)$$

$$\mathbf{b}_{\mathsf{jnt}}^{(r)} = \boldsymbol{\mu}_x^{(r)} - \mathbf{A}_{\mathsf{jnt}}^{(r)}\boldsymbol{\mu}_y^{(r)} \qquad (6)$$

$$\boldsymbol{\Sigma}_{\mathsf{b}}^{(r)} = \mathbf{A}_{\mathsf{jnt}}^{(r)}\boldsymbol{\Sigma}_y^{(r)}\mathbf{A}_{\mathsf{jnt}}^{(r)\mathsf{T}} - \boldsymbol{\Sigma}_x^{(r)} \qquad (7)$$

and for base-class $\mathtt{r}_r$ the joint clean, $\boldsymbol{x}_t$, and corrupted, $\boldsymbol{y}_t$, speech distribution is assumed to be Gaussian of the form

$$\begin{bmatrix}\boldsymbol{x}_t\\ \boldsymbol{y}_t\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\mu}_x^{(r)}\\ \boldsymbol{\mu}_y^{(r)}\end{bmatrix},\begin{bmatrix}\boldsymbol{\Sigma}_x^{(r)} & \boldsymbol{\Sigma}_{xy}^{(r)}\\ \boldsymbol{\Sigma}_{yx}^{(r)} & \boldsymbol{\Sigma}_y^{(r)}\end{bmatrix}\right) \qquad (8)$$

The form of the JUD compensation parameters is highly dependent on the covariance matrix structure in equation 8. If a full matrix is used then both $\mathbf{A}_{\mathsf{jnt}}^{(r)}$ and $\boldsymbol{\Sigma}_{\mathsf{b}}^{(r)}$ will be full. The transformation of the features for each of the base-classes can be cached. Thus the matrix-vector multiplication need only be performed once for each base-class. However the full variance bias results in a full covariance matrix for each Gaussian component. This increases the likelihood calculation cost from $\mathcal{O}(n)$ to $\mathcal{O}(n^2)$ for an n-dimensional feature vector. In addition, even when using a diagonal transform, the variance bias must be added to each of the variances of the recognition models. As one of the aims of uncertainty decoding is to decouple the compensation cost from the complexity of the recogniser, it would be preferable to remove this requirement.

3. PREDICTIVE LINEAR TRANSFORMS

Full JUD transforms have shown good performance compared to diagonal ones [6], but at a significantly increased computational load. Two linear transform approaches will be considered to address this computational issue. Both use the statistics obtained from a full JUD transform in the *transformed-space* specified by $\mathbf{A}_{\mathsf{jnt}}^{(r)}$ to estimate transforms. For component $\mathtt{s}_m$ in base-class $\mathtt{r}_r$, using equation 4,

$$\mathcal{E}\left\{\tilde{\boldsymbol{y}}_t\tilde{\boldsymbol{y}}_t^{\mathsf{T}}|\mathtt{s}_m\right\} = \boldsymbol{\Sigma}_{\mathsf{diag}}^{(m)} + \boldsymbol{\Sigma}_{\mathsf{b}}^{(r)} + \boldsymbol{\mu}^{(m)}\boldsymbol{\mu}^{(m)\mathsf{T}} \qquad (9)$$

$$\mathcal{E}\left\{\tilde{\boldsymbol{y}}_t|\mathtt{s}_m\right\} = \boldsymbol{\mu}^{(m)} \qquad (10)$$

$$\text{cov}(\tilde{\boldsymbol{y}}_t|\mathtt{s}_m) = \boldsymbol{\Sigma}_{\mathsf{diag}}^{(m)} + \boldsymbol{\Sigma}_{\mathsf{b}}^{(r)} \qquad (11)$$

where cov() yields the covariance matrix, and $\tilde{\boldsymbol{y}}_t$ is the JUD transformed observations

$$\tilde{\boldsymbol{y}}_t = \mathbf{A}_{\mathsf{jnt}}^{(r)}\boldsymbol{y}_t + \mathbf{b}_{\mathsf{jnt}}^{(r)} \qquad (12)$$

(ignoring the dependence of the transformed observation on the base-class for clarity). In addition the "occupancy" count for each component, $\gamma^{(m)}$, will be required. Since the transforms are meant to model the effects of the noise on the clean speech models, this occupancy is obtained from the training data. Thus all components will have non-zero occupancy counts. For this work the base-classes used for the adaptation transforms and precision matrix modelling are assumed to be the same as the JUD base-classes, though this is not a necessary constraint.

3.1. Adaptation-Based Transforms

In the adaptation-based transforms the underlying component specific model parameters are not altered, though the parameters may be transformed. Note predictive mean Maximum Likelihood Linear Regression (MLLR) [13] is not considered since, from equation 10, this will simply yield an identity transformation.

Constrained MLLR: is a linear transformation of the feature-space. For a component $\mathtt{s}_m$ belonging to base-class $\mathtt{r}_r$, the likelihood using a CMLLR transform [7] is (note $\tilde{\boldsymbol{y}}_t$ is defined in equation 12)

$$p(\boldsymbol{y}_t|\mathtt{s}_m) = |\mathbf{A}^{(r)}||\mathbf{A}^{(r)}_{\mathsf{jnt}}|\mathcal{N}\left(\mathbf{A}^{(r)}\tilde{\boldsymbol{y}}_t + \mathbf{b}^{(r)}; \boldsymbol{\mu}^{(m)}, \boldsymbol{\Sigma}^{(m)}_{\mathsf{diag}}\right) \quad (13)$$

An interesting attribute of this form of transformation is that both the component means and variances are unaltered. Thus the only computational cost in applying the transform is at decoding time. The estimation of the CMLLR transform in [7] may be expressed in terms of the predicted statistics

$$\begin{aligned}\mathbf{G}^{(r)}_i &= \sum_{m\in\mathtt{r}_r}\frac{\gamma^{(m)}}{\sigma^{(m)2}_{\mathsf{diag}i}}\left[\begin{array}{cc}\mathcal{E}\left\{\tilde{\boldsymbol{y}}_t\tilde{\boldsymbol{y}}_t^{\mathsf{T}}|\mathtt{s}_m\right\} & \mathcal{E}\left\{\tilde{\boldsymbol{y}}_t|\mathtt{s}_m\right\}\\ \mathcal{E}\left\{\tilde{\boldsymbol{y}}_t|\mathtt{s}_m\right\}^{\mathsf{T}} & 1\end{array}\right] \quad (14)\\ \mathbf{k}^{(r)}_i &= \sum_{m\in\mathtt{r}_r}\frac{\gamma^{(m)}\mu^{(m)}_i}{\sigma^{(m)2}_{\mathsf{diag}i}}\left[\begin{array}{c}\mathcal{E}\left\{\tilde{\boldsymbol{y}}_t|\mathtt{s}_m\right\}\\ 1\end{array}\right] \quad (15)\end{aligned}$$

Given these statistics the estimation proceeds in exactly the same fashion as standard CMLLR transform estimation (these are not reproduced here to save space, see [7] for details).

Full Covariance Transform: the likelihood for the MLLRCov transform is given by [7]

$$p(\boldsymbol{y}_t|\mathtt{s}_m) = |\mathbf{A}^{(r)}||\mathbf{A}^{(r)}_{\mathsf{jnt}}|\mathcal{N}\left(\mathbf{A}^{(r)}\tilde{\boldsymbol{y}}_t; \mathbf{A}^{(r)}\boldsymbol{\mu}^{(m)}, \boldsymbol{\Sigma}^{(m)}_{\mathsf{diag}}\right) \quad (16)$$

Compared to the CMLLR transform, the mean is transformed as well as the feature-space. As a by-product of this there is no transform bias as, by definition, the ML-estimate of the transformed mean is $\mathbf{A}^{(r)}\boldsymbol{\mu}^{(m)}$. Again the statistics for estimating the transform may be expressed in terms of predicted statistics

$$\begin{aligned}\mathbf{G}^{(r)}_i &= \sum_{m\in\mathtt{r}_r}\frac{\gamma^{(m)}}{\sigma^{(m)2}_{\mathsf{diag}i}}\text{cov}(\tilde{\boldsymbol{y}}_t|\mathtt{s}_m)\\ &= \sum_{m\in\mathtt{r}_r}\frac{\gamma^{(m)}}{\sigma^{(m)2}_{\mathsf{diag}i}}\left(\boldsymbol{\Sigma}^{(m)}_{\mathsf{diag}} + \boldsymbol{\Sigma}^{(r)}_{\mathsf{b}}\right) \quad (17)\end{aligned}$$

The transform is then estimated in the standard style using an iterative approach where

$$\mathbf{a}^{(r)}_i = \mathbf{c}_i\mathbf{G}^{(r)\text{-}1}_i\sqrt{\left(\frac{\sum_{m\in\mathtt{r}_r}\gamma^{(m)}}{\mathbf{c}_i\mathbf{G}^{(r)\text{-}1}_i\mathbf{c}^{\mathsf{T}}_i}\right)} \quad (18)$$

where $\mathbf{a}^{(r)}_i$ is the i^{th} row-vector of the transform $\mathbf{A}^{(r)}$ and $\mathbf{c}_i$ is the cofactor row-vector of $\mathbf{A}^{(r)}$. This MLLRCov requires that the mean is transformed, in addition to transforming the feature-space. This will have computational cost of $\mathcal{O}(Mn^2)$, where M is the number of components in the recognition system.

3.2. Structured Precision Matrix Transforms

The linear transforms in the previous section have only indirectly modelled the correlation for each component, as the covariance matrix was constrained to be the same as the original clean model. If this restriction is loosened then structured precision matrices can be used. Many of these may be described in a basis superposition framework [9]. Here

$$\boldsymbol{\Sigma}^{(m)\text{-}1} = \sum_{i=1}^{B}\lambda^{(m)}_i\mathbf{H}^{(i)} \quad (19)$$

where $\mathbf{H}^{(i)}$ is a basis matrix for modelling the precision matrices. Semi-tied covariance matrix [8] are examined in this paper, where $B = n$ and $\mathbf{H}^{(i)}$ is symmetric with rank 1. Here

$$\boldsymbol{\Sigma}^{(m)\text{-}1} = \mathbf{A}^{(r)\mathsf{T}}\tilde{\boldsymbol{\Sigma}}^{(m)\text{-}1}_{\mathsf{diag}}\mathbf{A}^{(r)} \quad (20)$$

and $\tilde{\boldsymbol{\Sigma}}^{(m)}_{\mathsf{diag}}$ is a diagonal covariance matrix. The likelihood can be calculated as

$$p(\boldsymbol{y}_t|\mathtt{s}_m) = |\mathbf{A}^{(r)}||\mathbf{A}^{(r)}_{\mathsf{jnt}}|\mathcal{N}\left(\mathbf{A}^{(r)}\tilde{\boldsymbol{y}}_t; \mathbf{A}^{(r)}\boldsymbol{\mu}^{(m)}, \tilde{\boldsymbol{\Sigma}}^{(m)}_{\mathsf{diag}}\right) \quad (21)$$

This is efficient as the transformed features can again be cached for each base-class, and then the decoding cost is the standard diagonal covariance matrix Gaussian calculation. The estimation of the transform, $\mathbf{A}^{(r)}$, and diagonal covariance matrix, $\tilde{\boldsymbol{\Sigma}}^{(m)}_{\mathsf{diag}}$, is an iterative process [8].

1. **Initialise** using

$$\mathbf{A}^{(r)} = \mathbf{I}, \quad \tilde{\boldsymbol{\Sigma}}^{(m)}_{\mathsf{diag}} = \text{diag}\left(\boldsymbol{\Sigma}^{(m)}_{\mathsf{diag}} + \boldsymbol{\Sigma}^{(r)}_{\mathsf{b}}\right) \quad (22)$$

2. **Update transform**, $\mathbf{A}^{(r)}$, using equation 18 with

$$\mathbf{G}^{(r)}_i = \sum_{m\in\mathtt{r}_r}\frac{\gamma^{(m)}}{\tilde{\sigma}^{(m)2}_{\mathsf{diag}i}}\left(\boldsymbol{\Sigma}^{(m)}_{\mathsf{diag}} + \boldsymbol{\Sigma}^{(r)}_{\mathsf{b}}\right) \quad (23)$$

3. **Update diagonal covariance matrix** using

$$\tilde{\boldsymbol{\Sigma}}^{(m)}_{\mathsf{diag}} = \text{diag}\left(\mathbf{A}^{(r)}\left(\boldsymbol{\Sigma}^{(m)}_{\mathsf{diag}} + \boldsymbol{\Sigma}^{(r)}_{\mathsf{b}}\right)\mathbf{A}^{(r)\mathsf{T}}\right) \quad (24)$$

4. **Goto 2** unless converged, or maximum number of iterations.

It is possible to stop the estimation at various stages. Stopping at step (1) yields the simple diagonalisation of the variance bias, $\boldsymbol{\Sigma}^{(r)}_{\mathsf{b}}$, described in [6], referred to as a 0-iteration system. Stopping after step (2) yields a form similar to a MLLRCov transform, equation 16, but with the diagonalised bias variance added. This will be referred to as a $\frac{1}{2}$-iteration semi-tied update. If step (3) is also completed then this is a complete semi-tied update.

Compared to the adaptation-based predictive linear transforms this is computationally more expensive. Multiple accumulations of the statistics, $\mathbf{G}^{(r)}_i$, are required as well as multiple transform estimations. Approximations for this are discussed in section 4. In common with the MLLRCov transform the mean must be transformed, cost $\mathcal{O}(Mn^2)$, but the variance must also be transformed, at a cost $\mathcal{O}(Mn^2)$ per full iteration.

4. COMPUTATIONAL COST

An important consideration when using linear transforms for noise robustness is the computational cost. One of the motivations for using JUD is that it is computationally efficient. However if full JUD transforms are used this makes decoding computationally very expensive. The predictive linear transforms address this problem by allowing diagonal covariance matrices to be used in the GMMs. It is therefore important that the estimation and application of the predictive linear transforms is as efficient as possible.

Statistic Accumulation: the adaptation predictive transform statistic accumulation can be made highly efficient compared to implementing equations 14 or 17. For example $\mathbf{G}^{(r)}_i$ in equation 17 can be expressed as

$$\mathbf{G}^{(r)}_i = \left(\sum_{m\in\mathtt{r}_r}\frac{\gamma^{(m)}\boldsymbol{\Sigma}^{(m)}_{\mathsf{diag}}}{\sigma^{(m)2}_{\mathsf{diag}i}}\right) + \boldsymbol{\Sigma}^{(r)}_{\mathsf{b}}\left(\sum_{m\in\mathtt{r}_r}\frac{\gamma^{(m)}}{\sigma^{(m)2}_{\mathsf{diag}i}}\right) \quad (25)$$

The first term on the right-hand-side is independent of the noise condition and thus can be accumulated and cached. Similarly for the elements in brackets in the second term on the right-hand-side. Thus the cost for accumulating statistics is simply $\mathcal{O}(n^3)$ for all n dimensions. Unfortunately this simple caching is not directly applicable to the structured precision matrices as from equation 23, $\mathbf{G}_i^{(r)}$ is function of the variance bias which depends on the noise condition.

Cofactor calculation: for all the schemes it is necessary to compute the cofactors of the current estimate of the transformation matrix. If implemented directly this has a cost of $\mathcal{O}(n^3)$. Consider the case for updating row i of the matrix $\mathbf{A}$ from $\mathbf{a}_i$ to $\mathbf{a}_i^{\texttt{new}}$. Using the Sherman-Morrison matrix inversion formula the inverse and determinant may be expressed as

$$\begin{aligned}\left(\mathbf{A}+\mathbf{e}_i(\mathbf{a}_i^{\texttt{new}}-\mathbf{a}_i)\right)^{\text{-}1} &= \mathbf{A}^{\text{-}1}-\frac{\mathbf{A}^{\text{-}1}\mathbf{e}_i(\mathbf{a}_i^{\texttt{new}}-\mathbf{a}_i)\mathbf{A}^{\text{-}1}}{1+(\mathbf{a}_i^{\texttt{new}}-\mathbf{a}_i)\mathbf{A}^{\text{-}1}\mathbf{e}_i} \\ |\mathbf{A}+\mathbf{e}_i(\mathbf{a}_i^{\texttt{new}}-\mathbf{a}_i)| &= \left(1+(\mathbf{a}_i^{\texttt{new}}-\mathbf{a}_i)\mathbf{A}^{\text{-}1}\mathbf{e}_i\right)|\mathbf{A}|\end{aligned}$$

where $\mathbf{e}_i$ is a zero column vector other than the i^{th} element which is set to one. These updates have computational cost $\mathcal{O}(n^2)$. The determinant can be simply calculated from the determinant lemma at no additional cost. The cofactors can then be computed using the equality

$$\mathbf{A}^{\text{-}1} = \left[\begin{array}{ccc}\mathbf{c}_1^{\mathsf{T}} & \dots & \mathbf{c}_n^{\mathsf{T}}\end{array}\right]^{\mathsf{T}}/|\mathbf{A}| \tag{26}$$

This is an exact calculation so will have no impact on recognition performance (ignoring possible numerical accuracy issues).

Matrix inversion: For the estimation of the transform it is necessary to invert the accumulated statistics $\mathbf{G}_i^{(r)}$ in equation 17 for example for each dimension. A modified version of Least Squares Linear Regression (LSLR) is used to make this more efficient. Here the occupancy weighted average within class covariance matrix is used for each base-class,

$$\overline{\boldsymbol{\Sigma}}_{\mathsf{diag}}^{(r)} = \left(\sum_{m\in\mathtt{r}_r}\gamma^{(m)}\boldsymbol{\Sigma}_{\mathsf{diag}}^{(m)}\right)\Bigg/\left(\sum_{m\in\mathtt{r}_r}\gamma^{(m)}\right) \tag{27}$$

This is used as an approximate target covariance matrix. Thus

$$\mathbf{G}_i^{(r)} \approx \frac{1}{\overline{\sigma}_{\mathsf{diag}i}^{(r)2}}\left(\sum_{m\in\mathtt{r}_r}\gamma^{(m)}\left(\boldsymbol{\Sigma}_{\mathsf{diag}}^{(m)}+\boldsymbol{\Sigma}_{\mathsf{b}}^{(r)}\right)\right) \tag{28}$$

The inversion only needs to be performed once (rather than once per dimension). This has cost $\mathcal{O}(n^3)$ per transform. When this form of approximation is used with the semi-tied transform, it is similar to the class specific decorrelating transforms in [14], though now predicted using a JUD transform. In addition to simplifying the inversion of $\mathbf{G}_i^{(r)}$ the accumulation of the statistics for the semi-tied transform is more efficient. Equation 23 can be re-written as

$$\mathbf{G}_i^{(r)} \approx \frac{1}{\overline{\tilde{\sigma}}_{\mathsf{diag}i}^{(r)2}}\left(\sum_{m\in\mathtt{r}_r}\gamma^{(m)}\boldsymbol{\Sigma}_{\mathsf{diag}}^{(m)}\right)+\frac{\boldsymbol{\Sigma}_{\mathsf{b}}^{(r)}}{\overline{\tilde{\sigma}}_{\mathsf{diag}i}^{(r)2}}\left(\sum_{m\in\mathtt{r}_r}\gamma^{(m)}\right) \tag{29}$$

where

$$\overline{\tilde{\boldsymbol{\Sigma}}}_{\mathsf{diag}}^{(r)} = \overline{\boldsymbol{\Sigma}}_{\mathsf{diag}}^{(r)}+\text{diag}\left(\boldsymbol{\Sigma}_{\mathsf{b}}^{(r)}\right) \tag{30}$$

Thus the weighted covariance matrix can be cached and used for all noise conditions. This will be referred to as the approximate estimation scheme.

Summary: a summary of the computational cost for estimating the transform and then applying the transform to compensate the models is given. For this section: M is the number of components in the recogniser, R is the number of base-classes, I is the number of updates of each row of the transform, and n is the dimensionality of the feature-vector. The cost of computing the full JUD transforms is not included, but the cost of applying the transform is included.

System	Statistics	Estimation
PST J-iter	$\mathcal{O}(J(Mn^2+Rn^3))$	$\mathcal{O}(JR(In^3+n^4))$
PST $\frac{1}{2}$-iter	$\mathcal{O}(Mn^2+Rn^3)$	$\mathcal{O}(R(In^3+n^4))$
PST Approx	$\mathcal{O}(Rn^2)$	$\mathcal{O}(Rn^3)$
PCMLLR	$\mathcal{O}(Rn^3)$	$\mathcal{O}(R(In^3+n^4))$
-Approx	$\mathcal{O}(Rn^2)$	$\mathcal{O}(Rn^3)$
PMLLRCov	$\mathcal{O}(Rn^3)$	$\mathcal{O}(R(In^3+n^4))$

Table 1. Predictive transform estimation cost, accumulation of statistics and transform estimation.

Table 1 shows the computational cost of estimating each of the predictive transforms. For the majority of situation $M \gg R$. As expected the most expensive scheme is the J-iteration Predictive Semi-Tied schemes (PST). Its cost can be compared to model-based schemes such as VTS and PMC which have cost $\mathcal{O}(Mn^3)$.

System	Features	Means	Variances
PST J-iter/Approx	$\mathcal{O}(RTn^2)$	$\mathcal{O}(Mn^2)$	$\mathcal{O}(Mn^2)$
PST $\frac{1}{2}$-iter	$\mathcal{O}(RTn^2)$	$\mathcal{O}(Mn^2)$	$\mathcal{O}(Mn)$
PCMLLR/Approx	$\mathcal{O}(RTn^2)$	—	—
PMLLRCov	$\mathcal{O}(RTn^2)$	$\mathcal{O}(Mn^2)$	—

Table 2. Compensation cost for predictive transform (T is the number of frames to recognise).

Table 2 shows the cost of applying each of the transforms. Again the most expensive scheme is the PST scheme. The cheapest are the PCMLLR and approximated PCMLLR (labelled PCMLLR/Approx) schemes as the complexity is not a function of the number of recogniser components. In addition, the Gaussian component likelihoods need to be computed. For all the predictive transform schemes, this will be $\mathcal{O}(TMn)$ compared to the full JUD scheme of $\mathcal{O}(TMn^2)$ for T frames of data.

5. EXPERIMENTS

Preliminary experiments for predictive linear transforms were conducted on two tasks. The first, AURORA 2, is used to illustrate the issue when there are low SNRs, but a simple task. The second database is a noise corrupted version of Resource Management. This allows the performance to be assessed on a medium vocabulary task with more complicated acoustic models. For these preliminary experiments the joint distribution for JUD was estimated using stereo data. This is not a requirement as these parameters may be estimated from a small number of noisy observations [15, 16]. All linear transforms, semi-tied, MLLRCov and CMLLR, were full in all experiments.

5.1. AURORA 2

AURORA 2 is a small vocabulary digit string recognition task. Utterances are one to seven digits long based on the TIDIGITS database with noise artificially added. The clean training data comprises 8440 utterances from 55 male and 55 female speakers. For matched training, 422 sentences are provided for each of 16 conditions: 4 different SNRs ranging from 20 to 5 dB, and with the 4 different additive noise sources N1 to N4: subway, babble, car and exhibition hall. Each of the 16 conditions also has a test set of a 1001 sentences with 52 male and 52 female speakers. A 39 dimensional feature vector consisting of 12 MFCCs appended with unnormalised log energy, delta and delta-delta coefficients was used. The acoustic models were 16 emitting state whole word digit models, with 3 mixtures per state and silence and inter-word pause models. For this work, HTK version 3.4 was used, as opposed to the reference 2.2 version, resulting in very minor differences in the baseline performance.

System	SNR(dB)			
	20	15	10	5
Clean	4.6	12.2	31.1	59.2
Clean ST	6.3	17.0	37.4	65.0
SPR	1.9	2.8	5.0	11.4
SPR ST	1.5	2.4	3.9	8.5
SPR CMLLR	1.7	2.4	4.5	10.9
JUD Diag	2.5	3.8	7.3	16.6
JUD Full	2.0	2.8	4.2	9.9

Table 3. Performance of clean, single-pass-retrained (SPR) matched, semi-tied (ST) and CMLLR, and Joint Uncertainty Decoding (JUD) diagonal and full performance on AURORA 2 test set A, averaged across N1-N4, WER(%).

Table 3 shows the various baseline performance numbers on the AURORA task. The first block of results shows the performance of the baseline clean models using no noise compensation. As expected the performance of the clean system is poor. The use of semi-tied covariance modelling (Clean ST) degrades performance for all SNRs compared to the standard clean system, despite reducing the WER in clean conditions from 1.06% to 0.95%. This shows the changes in feature space correlation due to noise. The second block of results relates to matched systems built using single-pass retraining (SPR) [17]. This was used to generate a baseline compensated system, analogous to an ideal model-based compensation scheme in the standard MFCC feature-space such as PMC, a semi-tied system (ST) using 16 semi-tied transforms and finally 16 CMLLR transforms (using the same base-classes as the ST system). It is interesting to note that using the CMLLR transforms outperforms the standard SPR system. This SPR CMLLR system is similar to an idealised version of schemes such as POF [2]. This is an example of where a standard model-based compensation scheme would not perform as well as a feature compensation scheme. However if the semi-tied system is used, then the model-based scheme outperforms the feature-based scheme, showing the importance of covariance matrix modelling.

Table 3 also shows the JUD decoding performance, again with 16 base-classes, where stereo data is used to obtain the joint distribution. As expected with 16 transforms, the diagonal transform system performs worse than the SPR system (increasing the number of base-classes to 546, the number of components, would result in the same performance). However using the full transform yields better performance than both the SPR system and the SPR CMLLR system. As discussed in the previous section, JUD is an interesting method to consider for noise compensation as it may be used for rapid compensation and the transforms estimated from limited data. Hence, this full JUD system will be used to obtain the statistics for the predictive linear transforms.

System	# iter	SNR(dB)			
		20	15	10	5
PST	0	31.2	50.4	79.2	89.7
	$\frac{1}{2}$	2.1	2.9	5.4	10.1
	10	2.1	2.9	5.3	9.9
PCMLLR	—	1.8	2.6	5.0	11.3
PMLLRCov	—	2.0	2.9	5.7	11.9

Table 4. Predictive semi-tied (PST), CMLLR and MLLRCov performance on AURORA 2 test set A, averaged across N1-N4, WER(%).

Three forms of predictive linear transform were examined, predictive semi-tied transforms (PST), predictive CMLLR (PCMLLR) and predictive MLLRCov (PMLLRCov) transforms. The results for these on the AURORA task are shown in table 4. Three versions of PST were evaluated. As noted in previous work the 0-iteration PST yields poor performance [6]. The $\frac{1}{2}$ iteration scheme and the 10 iteration scheme show similar performance. Other than the 10dB SNR condition, the performance is similar to the JUD full system used to get the statistics. Both the PCMLLR and PMLLRCov are worse than the PST system at low SNR (5dB). This is as expected, since at low SNRs the model parameters will need to be compensated for best performance. Note, the PCMLLR system approximates the SPR CMLLR system, but using only the full JUD statistics.

System	Estimation	SNR(dB)			
		20	15	10	5
PST	Exact (10 iter)	2.1	2.9	5.3	9.9
	Approx	2.1	3.0	5.3	9.9
PCMLLR	Exact	1.8	2.6	5.0	11.3
	Approx	1.8	2.6	5.2	11.7

Table 5. Exact and Approximate matrix inversion for PCMLLR on AURORA 2 test set A, averaged across N1-N4, WER(%).

One of the issues with predictive linear transforms is the computational cost in estimating the transform. In particular for the PST systems where the statistics must be accumulated, as well as the transform applied. Table 5 shows the performance using the matrix inversion approximation from section 4. For the PST system (10 iterations) the effect of the approximation is very small, similarly for the PCMLLR system.

5.2. Resource Management

For this work, noise was artificially added to a medium vocabulary speech recognition task, the 1000 word Resource Management (RM) database. Operations Room noise from the NOISEX-92 database was added at the waveform level. Though this task is artificial and

[2] In [6] model-based CMLLR schemes are shown to outperform the equivalent front-end CMLLR scheme. Thus the SPR CMLLR scheme is an optimistic estimate of the performance of linear feature compensation schemes.

is expected to yield better performance than would be obtained on realistic data, it allows a comparison of the various techniques in a highly controlled fashion. RM was used as a speaker independent task which consists of 109 training speakers reading 3990 sentences, 3.8 hours of data. All results are quoted as an average of three of the four available test sets, Feb89, Oct89 and Feb91, a total of 30 test speakers and 900 utterances. State-clustered triphone models were built using the HTK RM recipe. 16 base-classes were again used for all transforms.

System	# iter	Avg
Clean	—	33.2
SPR	—	7.2
SPR ST	10	6.7
SPR CMLLR	—	8.9
JUD Diag	—	8.2
JUD Full	—	7.4
PST	$\frac{1}{2}$	7.6
	10	7.3
	Approx	7.6

Table 6. Clean, single-pass-retrained (SPR) matched and semi-tied (ST), and Joint Uncertainty Decoding (JUD) and predictive semi-tied (PST) performance on 20dB SNR corrupted RM, Feb89, Oct89 an Feb91, average WER (%).

Table 6 shows both the baseline performance figures and the predictive semi-tied performance averaged over the Feb89, Oct89 and Feb91 test sets. In contrast to the AURORA 2 task the SPR system out-performed the SPR CMLLR system. There are a number of reasons for this. The task is harder, so good modelling of the distributions is important. The ratio number of recogniser components to base-classes is approximately 17 times larger than for the AURORA task. Also the SNR is higher so the impact of the transform is expected to be less. The PST system outperformed both SPR ST and JUD diag systems. The PST approximation to the JUD full system was again good, with no degradation observed. This illustrates that predictive linear transforms work well even on more complex tasks.

6. CONCLUSIONS

This paper has examined correlation modelling for noise robust speech recognition. The correlation between the elements of the feature vector vary as the signal to noise ratio is changed. As diagonal covariance matrix GMMs are used to model the HMM state-output distributions, correctly handling the changes in the feature vector correlations should improve recognition performance. JUD with full transforms is one approach that allows these correlation changes to be modelled, but makes the decoding process expensive. To address this problem approximations to the full JUD transforms are proposed. Two approaches are described in this paper, both under the general heading predictive linear transforms. These transforms may either be in the form of adaptation transforms, such as CMLLR or MLLR-Cov, or precision matrix models as in semi-tied covariance models. In contrast to the normal estimation schemes used, the transforms are estimated on statistics derived from the full JUD transforms, which can be made computationally efficient. Two databases were used for initially assessing the performance of these predictive linear transforms, AURORA 2 and a noise corrupted version of the Resource Management task. On both tasks the use of predictive linear transforms gave good noise robustness.

The current experiments use stereo data to estimate the JUD transform parameters. Future work will use the estimation approaches described in [16]. In addition the approaches will be applied to data recorded in low SNR conditions, rather than using artificially corrupted clean data.

7. REFERENCES

[1] M.J.F. Gales and S.J.Young, "Robust continuous speech recognition using parallel model combination," *IEEE Trans. on Speech and Audio Processing*, 1996.

[2] P. Moreno, *Speech Recognition in Noisy Environments*, Ph.D. thesis, Carnegie Mellon University, 1996.

[3] L. Neumeyer and M. Weintraub, "Probabilistic optimum filtering for robust speech recognition," in *Proc. ICASSP*, 1994, vol. 1, pp. 417–420.

[4] J. Droppo and A. Acero, "Joint discriminative front end and back end training for improved speech recognition accuracy," in *Proc. ICASSP*, Toulouse, France, 2006.

[5] J. Droppo, A. Acero, and L. Deng, "Uncertainty decoding with SPLICE for noise robust speech recognition," in *Proc. ICASSP*, Orlando, Florida, May 2002.

[6] H. Liao and M.J.F. Gales, "Joint uncertainty decoding for noise robust speech recognition," in *Proc. Interspeech*, Lisbon, Portugal, Sept. 2005.

[7] M.J.F. Gales, "Maximum Likelihood Linear Transformations For HMM-Based Speech Recognition," *Computer Speech and Language*, vol. 12, Jan. 1998.

[8] M.J.F Gales, "Semi-tied covariance matrices for hidden Markov models," *IEEE Transactions Speech and Audio Processing*, vol. 7, pp. 272–281, 1999.

[9] K.C. Sim and M.J.F. Gales, "Basis superposition precision matrix modelling for large vocabulary continuous speech recognition," in *Proc. ICASSP*, 2004.

[10] P. Olsen and R.A. Gopinath, "Modelling inverse covariance matrices by basis expansion," in *Proc. ICASSP*, 2002.

[11] M.J.F. Gales, "Predictive model based compensation schemes for robust speech recognition," *Speech Communication*, vol. 25, 1998.

[12] H. Liao and M.J.F. Gales, "Issues with uncertainty decoding for noise robust speech recognition," in *Proc. Interspeech*, 2006.

[13] C. Leggetter and P.C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density HMMs.," *Computer Speech and Language*, vol. 9, pp. 171–186, 1995.

[14] A Ljolje, "The importance of cepstral parameter correlations in speech recognition," *Computer Speech and Language*, vol. 8, pp. 223–232, 1994.

[15] H. Xu, L. Rigazio, and D. Kryze, "Vector Taylor series based joint uncertainty decoding," in *Proc. Interspeech*, 2006.

[16] H. Liao and M.J.F. Gales, "Adaptive training with joint uncertainty decoding for robust recognition of noisy data," in *Proc. ICASSP*, 2007.

[17] S.J. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P.C. Woodland, *The HTK Book (for HTK Version 3.4)*, University of Cambridge, Dec. 2006.