A TEACHING AND EVALUATION TOOL FOR ADAPTIVE SIGNAL PROCESSING USING JAVA

*M Harteneck** *R W Stewart*

Signal Processing Division, Dept. of Electrical and Electronic Engineering, University of Strathclyde, Glasgow, G1 1XW, Scotland, U.K., email: moritz@spd.eee.strath.ac.uk, bob@spd.eee.strath.ac.uk.

ABSTRACT

In this paper, a teaching and evaluation tool for adaptive algorithms using the JAVA platform is presented. The tool has been developed for use in teaching adaptive signal processing and gives the students the facility to observe a comprehensive set of algorithms executing in the time, frequency and z-domain, vary any parameters and thereby augment the traditional learning process. Another key aim in the development was to provide a simple tool so that the feasibility of adaptive algorithms for a particular problem can be evaluated quickly. The JAVA platform has been chosen for this task since it is possible to run the tool on any computer system (e.g. Unix, Windows, Linux) using a JAVA virtual machine via the world wide web. The tool is freely available to use at *http://www.spd.eee.strath.ac.uk/users /moritz/algorithmDemo/jdk1.1*.

1. INTRODUCTION

Adaptive signal processing is one of the most important classes of algorithms for modern DSP and communication systems. For example, the ubiquitous modem, now running easily at rates of 28800 baud can achieve this high rate as a result of adaptive echo cancelation and adaptive equalization algorithms [1]. The new generation of mobile multimedia systems and set-top boxes will also require extensive adaptive DSP. Similarly, the audio subsystem for video teleconferencing requires acoustic echo cancelation to be performed; once again this is primarily enabled by adaptive digital signal processing algorithms [2]. Even the technology of active noise cancelation is now implemented in the cabins of some airliners is enabled by adaptive signal processing.

The multitude of possible applications and the relative matureness of adaptive algorithms has created a growing demand for formal courses being taught on the subject. While teaching Master's level courses at the University of Strathclyde over the last 7 years, it was noticed that many students could understand and derive the adaptive algorithms [3, 4, 5] quite easily but when it came to evaluate their performance and decide which one to use, the ability to interpret the necessary equations and performance criteria describing the algorithms was missing. Therefore, it was decided to develop a teaching tool which would demonstrate how adaptive algorithms adapt to their solution, allow a comparison between different algorithms and install an intuitive understanding for the effects caused by varying various algorithm and system parameters. Previous attempts

*M Harteneck is now with Siemens AG, Munich.

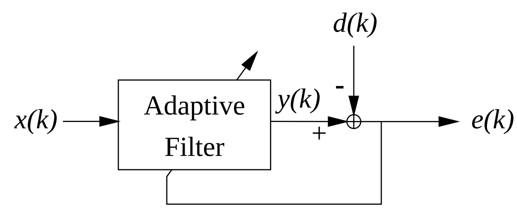

Figure 1. General Adaptive Signal Processor.

to produce effective teaching software were based around Windows 3.1 and C++ programming [6].

Another area where a collection of easily usable adaptive algorithms is of value is for engineers who want to rapidly evaluate whether an adaptive algorithm would bring an advantage in their particular application [7] while not wishing to intensively study adaptive filtering theory or expend a great effort in implementations to while encountering all the problems of instability and programming errors.

The platform chosen for the tool was the JAVA programming environment (*http://java.sun.com*) which is independent of the underlying computer system, i.e. any application written in this language can be downloaded easily via the internet and run on every machine which has a JAVA interpreter or a JAVA just-in-time compiler. Another advantage of this platform is that the programs can be integrated in a page on the world wide web (WWW) as so called "applets" and therefore students can use the tool on any computer, at university or at home, which is connected to the WWW. Hence problems such as machine availability, location and licensing are virtually eradicated.

In the remainder of this paper, first, adaptive signal processing is reviewed briefly and the major techniques are outlined. Then, in Section 3., the teaching and evaluation tool is presented and some screenshots are shown. Finally, in Section 4. some conclusions are drawn.

2. ADAPTIVE SIGNAL PROCESSING

Adaptive signal processing applications can be classified into four main groups: (a) noise cancelation, (b) system identification, (c) inverse system identification and (d) prediction [4]. The common element in all these groups is the general adaptive signal processor, as depicted in Fig. 1, with the input signal $x(k)$, the output signal $y(k)$, the desired signal $d(k)$ and the error signal $e(k)$. The aim of the adaptive algorithm is to minimise the power of the error signal and thereby make the desired signal $d(k)$ and the output signal $y(k)$ "similar". The internal structure of this general adaptive processor can vary and is application dependent. Usually finite impulse response (FIR) filters are used due to

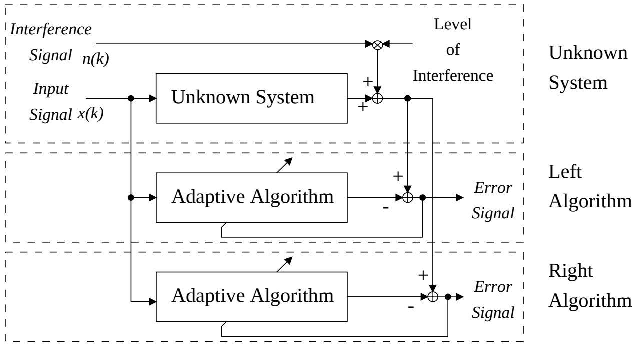

Figure 2. Internal Signal Flow of the Applet.

their inherent stability and the simplicity of the resulting algorithms [3], however, the field of adaptive IIR algorithms is an active area of research [8] where the algorithms can be based on direct form and lattice form implementations. To minimize the power of the error signal $e(k)$, two different approaches are usually taken: (1) to minimize a mean squared error measure and (2) to minimize the sum of squared errors measure. The first approach leads to the family of least mean squares (LMS) algorithms and the second leads to the family of least squares (LS) algorithms.

For further information on the adaptive techniques, the reader is referred to [3, 9, 5, 8, 10, 4].

3. TEACHING TOOL

When teaching the formulation and performance attributes of adaptive algorithms in a class environment, it has been noted that the students can easily derive and understand the equations presented, but they showed problems of understanding how the algorithms actually "adapt" and found it difficult to tell which one converges in which way and why. For this reason, a JAVA applet has been developed which gives the student a tool to learn about the algorithm by easy interactive experiments and observations.

Another motivation for the tool is to give engineers the possibility to rapidly evaluate the performance of standard adaptive techniques without having to know too much about adaptive signal processing by giving them the possibility to feed user-supplied input and desired signals into the tool and analyze the error signals without having to worry about programming mistakes or other problems often encountered in the usual learning process.

3.1. Description

Fig. 2 shows the internal signal flow of the applet. The *Unknown System* generates the input and the desired signals according to the input and interference source selected and the unknown system chosen. When using the tool, the user has three choices for the unknown system:

Finite Impulse Response System: The unknown system is an FIR system. The system is represented in two ways; first the impulse response which the user can change by mouse clicks, and a second representation which can be altered between the magnitude of the transfer function and the zeros of the system.

Infinite Impulse Response System: The unknown system is an IIR model where the system can be changed by either dragging the poles and zeros in the z-domain, by adding or removing poles and zeros or by changing the feedforward and feedback weights directly. When the feedback path of the system is changed stability is always ensured. Other representations of the system are the magnitude of the transfer function and its impulse response.

User Supplied Data Files: In this case the user can supply two data files as input and desired signal and store the error signals into separate files. Note that due to security restrictions which are inherent to the JAVA platform, this option is only available if the system is run locally as an application and not as an applet inside a browser.

Table 1. Implemented Source Types.

	Source Type
1	White Noise
2	White Noise with Slowly Changing Variance
3	Noise with a Low Pass Characteristic
4	Noise with a Band Pass Characteristic
5	Noise with a High Pass Characteristic
6	Sinusoid with a Low Frequency
7	Sinusoid with a Medium Frequency
8	Sinusoid with a High Frequency

Table 2. Implemented Adaptive Algorithms.

System	Family	Algorithm
FIR	LMS	Least Mean Squares [3]
		Normalized Least Mean Squares
		Sign-Data
		Sign-Error
		Sign-Error-Sign-Data
	LS	QR Decomposition
		Recursive Least Squares [3]
		Random Walk
IIR	LMS	Feintuch [11]
		Simplified Gradient [12]
		Full Gradient [12]
	LS	QR Decomposition [13]

For the FIR and IIR unknown system, eight different source types are available to model the input signal $x(k)$ and the interference signal $n(k)$. These source types are listed in Tab. 1 and cover the most important classes of sources which are needed to understand adaptive algorithms. Note that the tool also allows the user to input any signal directly from a file.

The two adaptive algorithms can be chosen out of the twelve algorithms which are shown in Tab. 2. For each algorithm a representation of the actual state of the adaptive filter and the error signal for the last 100 samples are shown. The representation of the state can be chosen to be for FIR adaptive filters the impulse response, the magnitude response or the zeros of the transfer function and for IIR adaptive filters to be the impulse response, the feedforward weights, the feedback weights, the magnitude response or the pole-zero plot of the transfer function. The error signal can be represented either in the time domain or in the frequency domain.

3.2. Screenshots

In this section some typical screenshots of the tool are shown to give an impression how the application operates.

Fig. 3 shows the initial configuration of the tool. In the menubar the user can select the unknown system, choose the algorithm for the left and right adaptive filter and select the sampling frequency varying from 0.5 Hz to as fast

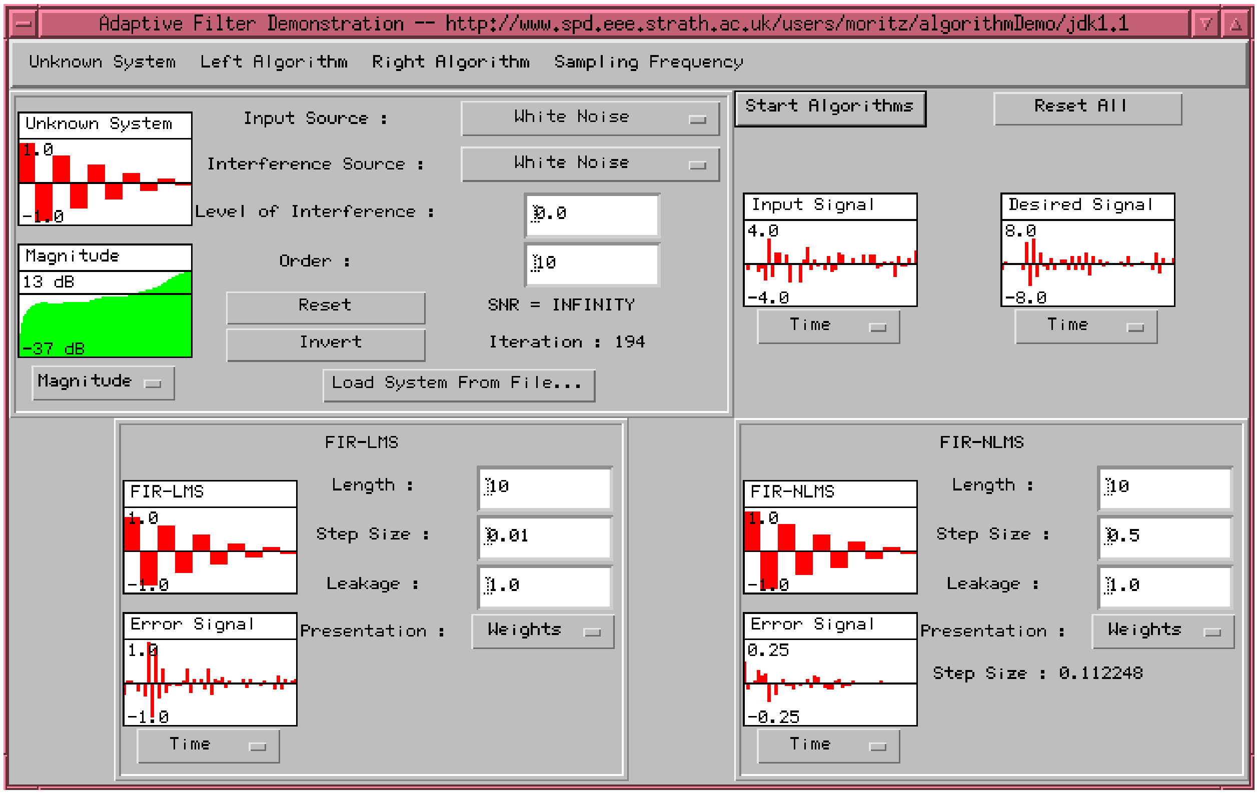

Figure 3. Window simulating two adaptive FIR algorithms for an FIR unknown system.

as possible. In the top left-hand corner the unknown system, in this case an FIR system, is shown, where the unknown system is represented by its impulse and magnitude response. The two choices allow the user to select among the different signal sources. The level of the interference source and the order of the unknown system can be chosen below. To provide information on the power of the interference, the signal-to-noise ratio is calculated at every time step and the button at the bottom allows the user to load an unknown system from disk. Pressing the `RESET` button will bring the unknown system to a defined state and pressing the `INVERT` button will multiply all the weights by -1 to give an abrupt system change. To the right of the unknown system, the input signal and the desired signal are plotted and below the two adaptive algorithms, in this case an FIR-LMS and an FIR-NLMS algorithm, are operating in parallel to allow a visual comparison. The input source and the interference source are both chosen to be white noise. The actual state of the two algorithms is represented in both cases by the impulse response and the error signals are shown in their time representation.

Fig. 4 shows the tool where the unknown system is an IIR system, with the unknown system being represented by a pole-zero plot in the upper representation and by a magnitude plot in the lower representation. To change the unknown system, the user can either move the poles and zeros with the mouse or change the feedforward and feedback weights directly. To increase or reduce the order of the unknown system, the user has the possibility to add or remove poles and zeros via mouse clicks. The two adaptive algorithms, which are shown in the lower half of the window, are in this case Feintuch's IIR-LMS algorithm, with the impulse response and the frequency response as representations, and the IIR-QR algorithm with the pole-zero plot and the magnitude response as representations. The input source and the interference source are both chosen to be white noise.

4. CONCLUSIONS

In this paper we have presented a learning and evaluation tool to teach adaptive signal processing and to evaluate whether the use of adaptive techniques would offer an advantage in a particular application.

It has been noted during courses taught at the University of Strathclyde that students usually understand the mathematical basis for adaptive filtering algorithms very well but then lack the understanding (which comes through experience) of the actual performance of the algorithms. Another problem which was encountered was the fact that engineers understand the equations, which describe the adaptive filters, but then have problems to evaluate which algorithm, what step size, filter length, etc. to use in a particular application and to decide whether an adaptive technique offers an advantage over different techniques. This can be especially difficult for an engineer new to the field of adaptive signal processing. Therefore, a tool was developed which the engineers could use to quickly evaluate the performance of various algorithms to decide whether it is worthwhile to learn more about them for the particular problem.

This general tool has been developed to demonstrate to the student the actual performance of adaptive algorithms in time, frequency and z-domain and to make on-line experiments very easy. To the engineer the interface to load user-supplied files and perform the adaptive filtering task without actual implementation and programming problems

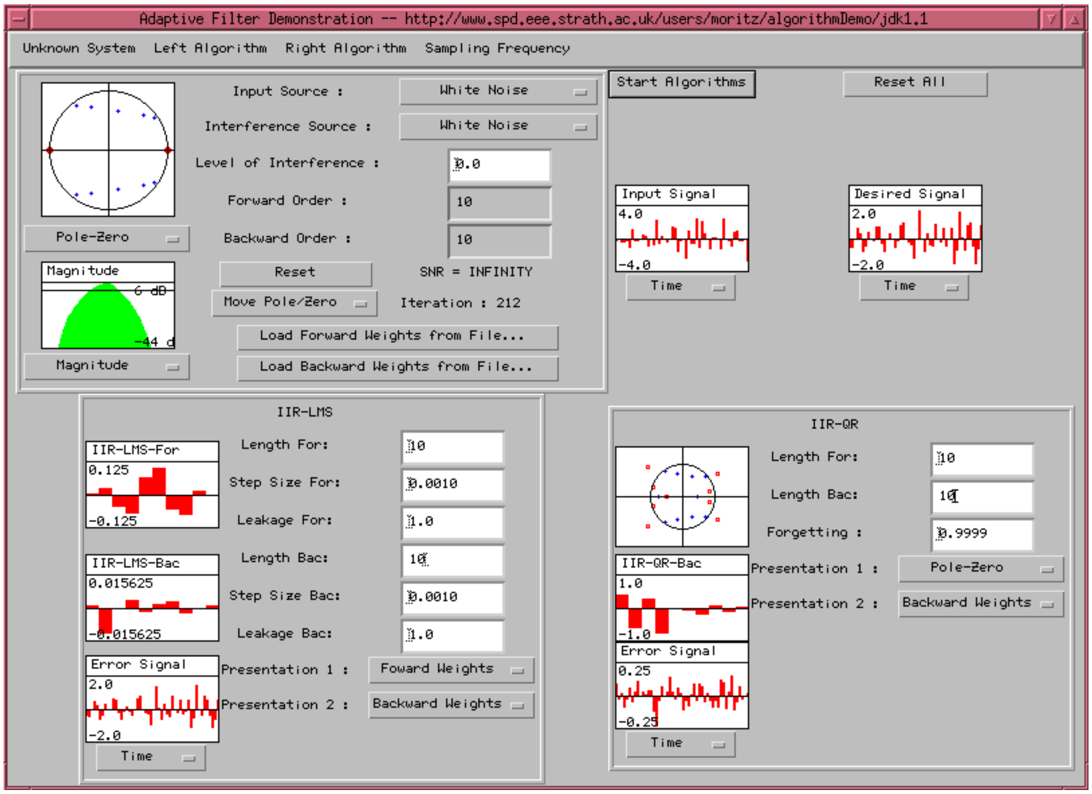

Figure 4. Window simulating two adaptive IIR algorithms and for an IIR unknown system.

can shorten the time to evaluate these strategies significantly.

The tool is programmed using the JAVA language to support multiple platform implementation and implements the most commonly used adaptive FIR and IIR algorithms. This paper has attempted to succinctly describe the general system but to fully evaluate the work we would suggest that readers run the tool as an applet or an application which is freely available over the world wide web together with some demostration files at *http://www.spd.eee.strath.ac.uk/users/moritz/algorithmDemo/jdk1.1*.

REFERENCES

[1] J Gibson, *Mobile Communications*, IEEE Press, 1995.

[2] E Hänsler, "The Hands-Free Telephone Problem: A Second Annotated Bibliography Update", *Proceedings of the International Workshop on Acoustic Echo and Noise Control, Roros, Norway*, June 1995.

[3] S Haykin, *Adaptive Filter Theory*, Prentice Hall Information and Systems Sciences Series, 3^{rd} edition, 1996.

[4] B Widrow and S D Stearns, *Adaptive Signal Processing*, Prentice Hall, Englewood Cliffs, 1985.

[5] O Macchi, *Adaptive Processing*, John Wiley & Sons, 1995, ISBN-0-471-93403-8.

[6] R W Stewart and P Aaserud, "Adaptlab: An Adaptive Signal Processing Environment for Teaching and Development", *Proceedings of 28th Asilomar Conference on Signals, Systems and Computers, Monterey, USA*, vol. 2, pp. 1320–1324, November 1994, 1994.

[7] R W Stewart, "Practical DSP for Scientists and Engineers", *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, Minneapolis, USA*, vol. 1, pp. 32–35, April 1993.

[8] P A Regalia, *Adaptive IIR Filtering in Signal Processing and Control*, Marcel Dekker, 1994.

[9] S T Alexander, *Adaptive Signal Processing, Theory and Applications*, Springer-Verlag, 1986, ISBN 3-540-96480-4.

[10] J J Shynk, "Adaptive IIR Filtering", *IEEE ASSP Magazine*, vol. 6, no. 4, pp. 4 – 21, April 1989.

[11] P L Feintuch, "An Adaptive Recursive LMS Filter", *Proceedings of the IEEE*, vol. 64, no. 11, pp. 1622 – 1624, November 1976.

[12] J J Shynk, "Adaptive IIR Filtering Using Parallel-Form Realizations", *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 4, pp. 519 – 532, April 1989.

[13] M Harteneck and R W Stewart, "A Fast Converging Algorithm for Landau's Output Error Method", *Journal of Applied Mathematics and Computer Science*, vol. 8, no. 2, 1998.