AN AUGMENTED ERROR CRITERION FOR LINEAR ADAPTIVE FILTERING: THEORY, ALGORITHMS AND APPLICATIONS

By

YADUNANDANA NAGARAJA RAO

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2004

This dissertation is dedicated to my family, teachers and friends for their enduring love, support and friendship.

ACKNOWLEDGMENTS

First of all, I would like to thank Dr. Jose Principe, for his constant guidance, encouragement, patience and continuous support over the past five years. His enthusiasm for research and quest for excellence have left an everlasting impression in my mind. To me, he has been more than an advisor, and this research would not have been possible without him. Secondly, I would like to thank Dr. John Harris for being on my committee and offering me guidance not only in research but in many other aspects of life. I would also like to thank Dr. Michael Nechyba and Dr. Mark Yang for being on my committee.

I would like to thank Dr. Deniz Erdogmus, my friend and colleague at CNEL, whose contributions in this research have been tremendous. I deeply benefited from all the long hours of fruitful discussions with him on a multitude of topics. His drive for research and enormous ability to motivate others have been quite inspirational. I also wish to extend my acknowledgements to all the members of CNEL who have been primarily responsible for my fruitful stay in the lab. I would like to extend my gratitude to the always cheerful Ellie Goodwin for her golden words of wisdom. Her ability to get things done was truly remarkable. I would also like to acknowledge Linda Kahila for the extensive support and assistance she provided during my stay at UFL.

I would like to thank my family and friends for their constant love and encouragement. They have allowed me to pursue whatever I wanted in life. Without their guidance and affection, it would have been impossible for me to advance my education.

Lastly, I would like to thank my life partner Geetha for making my life beautiful and for being on my side whenever I needed her. Her everlasting love has made me a better individual.

TABLE OF CONTENTS

page

LIST OF TABLES

LIST OF FIGURES

Figure page

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

AN AUGMENTED ERROR CRITERION FOR LINEAR ADAPTIVE FILTERING: THEORY, ALGORITHMS AND APPLICATIONS

By

Yadunandana Nagaraja Rao

May 2004

Chair: Jose C. Principe
Cochair: John G. Harris
Major Department: Electrical and Computer Engineering

Ever since its conception, the mean-squared error (MSE) criterion has been the workhorse of optimal linear adaptive filtering. However, it is a well-known fact that the MSE criterion is no longer optimal in situations where the data are corrupted by noise. Noise, being omnipresent in most of the engineering applications, can result in severe errors in the solutions produced by the MSE criterion. In this dissertation, we propose novel error criteria and the associated learning algorithms followed by a detailed mathematical analysis of these algorithms. Specifically, these criteria are designed to solve the problem of optimal filtering with noisy data. Firstly, we discuss a new criterion called augmented error criterion (AEC) that can provide unbiased parameter estimates even in the presence of additive white noise. Then, we derive novel, online sample-by-sample learning algorithms with varying degrees of complexity and performance that are tailored for real-world applications. Rigorous mathematical analysis of the new algorithms is presented.

In the second half of this dissertation, we extend the AEC to handle correlated noise in the data. The modifications introduced will enable us to obtain optimal, unbiased parameter estimates of a linear system when the data are corrupted by correlated noise. Further, we achieve this without explicitly assuming any prior information about the noise statistics. The analytical solution is derived and an iterative stochastic algorithm is presented to estimate this optimal solution.

The proposed criteria and the learning algorithms can be applied in many engineering problems. System identification and controller design problems are obvious areas where the proposed criteria can be efficiently used. Other applications include model-order estimation in the presence of noise and design of multiple local linear filters to characterize complicated nonlinear systems.

CHAPTER 1
MEAN SQUARED ERROR BASED ADAPTIVE SIGNAL PROCESSING SYSTEMS: A BRIEF REVIEW

Introduction

Conventional signal processing techniques can be typically formulated as linear or non-linear operations on the input data. For example, a finite impulse response (FIR) filter is a linear combination of the time delayed versions of the input signal. We know that a linear combiner is nothing but a linear projector in the input space. Mathematically speaking, a projection can be defined as a linear transformation between two vector spaces [1]. These linear transformations can be vectors spanning $\Re^{nx1}$ or matrices spanning $\Re^{nxn}$. For vector transformations the projections are given by the inner products and in case of matrix transformations the projections become rotations. Often, most of the design tasks in signal processing involve finding appropriate projections that perform the desired operation on the input. For instance, the filtering task is basically finding the projection that preserves only a specified part of the input information [2]. Another example is data compression, wherein we estimate an optimal projection matrix or rotation matrix that preserves most of the information in the input space. The first step in finding these projections is to understand the specifications of the problem. Then, the specifications are translated into mathematical criteria and equations that can be solved using various mathematical and statistical tools. The solutions thus obtained are often optimal with respect to the criterion used.

Why Do We Need Adaptive Systems?

Depending on the problem at hand, estimating the optimal projections can be a daunting task. Complexities can arise due to the non-availability of a closed form solution or even the non-existence of a feasible analytical solution. In the latter case, we may have to be contented with sub-optimal solutions. On the other hand, scenarios exist where we have to synthesize projections that are not based on user specifications. For instance, suppose we are given two signals, an input and a desired signal, and the goal is to find the optimal projection (filter) that generates the desired signal from the input. Thus the specifications do not convey any explicit information regarding the type of filter we have to design. The conventional filter synthesis cookbook does not contain any recipes for these types of problems. Such problems can be solved by learning mechanisms that intelligently deduce the optimal projections using only the input and desired signals or at times using the input signal alone. These learning mechanisms form the foundation of adaptive systems and neural networks. All learning mechanisms have at least two major pieces associated with them. The first is the criterion and the second is the search algorithm. The search algorithm finds the best possible solution in the space of the inputs under some constraints. Optimization theory has provided us with a variety of search techniques possessing different degrees of complexity and robustness [3]. These learning-based adaptive systems provide us with a powerful methodology that can go beyond conventional signal processing. The projections derived by these adaptive systems are called optimal adaptive projections. Another very desirable feature of adaptive systems is their innate ability to automatically adjust and track according to the changing statistical properties of signals. This can be vital in many engineering applications, viz., wireless data transmission, biomedical monitoring and control, echo cancellation over wired

telephone lines etc., wherein the underlying physical sources that generate the information change over time. In the next section, we will briefly review the theory behind the design of linear adaptive systems.

Design of Adaptive Systems

A block diagram of an adaptive system is shown in Figure 1-1. Assume that we are given a zero-mean input signal x_n and a zero-mean desired signal d_n. Further, these signals are assumed to be corrupted by noise terms v_n and u_n respectively. Let the parameters of the adaptive system be denoted by the weight vector **w**. Note that we have not put any constraints on the topology of the adaptive filter. For convenience, we will assume a FIR topology in this chapter. The goal then is to generate an output y_n that best approximates the desired signal. In order to achieve that, a criterion (often referred to as the *cost* $J(\mathbf{w})$) is devised which is typically a function of the error e_n defined as the difference between the desired signal and the output, i.e., $e_n = d_n - y_n$. The most widely used criterion in the literature is the Mean-Squared Error (MSE) which is defined as

$$J(\mathbf{w}) = E(e_n^2) \tag{1.1}$$

The MSE cost function has some nice properties, namely,

- Physical relevance to energy
- The performance surface (shape of $J(\mathbf{w})$) is smooth and has continuous derivatives
- The performance surface is a convex paraboloid with a single global minimum
- The weight vector $\mathbf{w}_*$ corresponding to the global minimum is the best linear unbiased estimate in the absence of noise [4]
- If the desired signal is a future sample of the input, i.e., $d_n = x_{n+\tau}$, then the filter with coefficients $\mathbf{w}_*$ is guaranteed to be minimum phase [5]

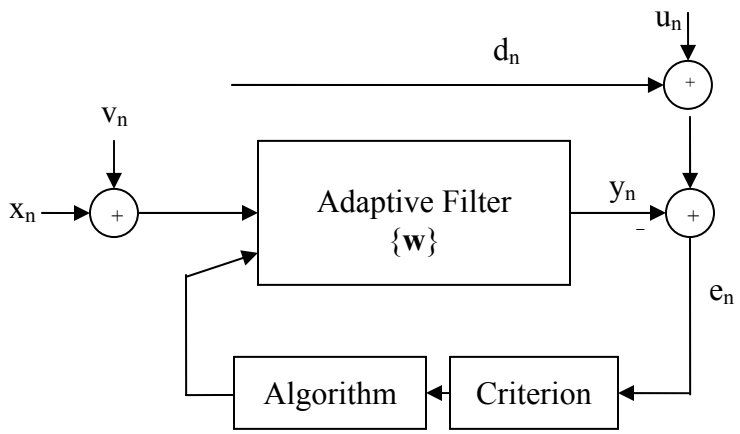


Figure 1-1. Block diagram of an Adaptive System.

Once the criterion is fixed, the next step is to design an algorithm to optimize the cost function. This forms another important element in an adaptive system. Optimization is a well researched topic and there is a plethora of search methods for convex cost functions. Specifically, we minimize the MSE cost function and since the performance surface is quadratic with a single global minimum, an analytical closed form optimal solution $\mathbf{w}_*$ can be easily determined. The optimal solution is called the Wiener solution for MSE [6] (Wiener filter), which is given by

$$\mathbf{w}_* = \mathbf{R}^{-1}\mathbf{P} \tag{1.2}$$

In equation (1.2), **R** denotes the covariance matrix of the input defined as $\mathbf{R} = E\left(\mathbf{x}_k \mathbf{x}_k^T\right)$ and the vector **P** denotes the cross correlation between the desired signal and the lagged input defined as $\mathbf{P} = E\left(\mathbf{x}_k d_k\right)$. Computing the Wiener solution requires inverting the matrix **R** which requires $O(N^3)$ operations [7]. However, due to the time-delay embedding of the input, the matrix **R** can be easily shown to be symmetric and Toeplitz, which facilitates a computationally efficient inverse operation with complexity $O(N^2)$ [8]. From the point of view of an adaptive system, the Wiener solution is still not elegant

because one requires the knowledge of all data samples to compute equation (1.2). A sample-by-sample (iterative) algorithm is more desirable as it suits the framework of an adaptive system. The most commonly used algorithms to iteratively estimate the optimal Wiener solution $\mathbf{w}_*$ are the stochastic gradient based Least Mean Squares (LMS) and the fixed-point type Recursive Least Squares (RLS).

Least Mean Squares (LMS) Algorithm

The gradient of the cost function in (1.1) is given by

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = -2E(e_k \mathbf{x}_k) \tag{1.3}$$

Notice that the output of the adaptive filter y_n is simply the inner-product between the weight vector $\mathbf{w}$ and the vector $\mathbf{x}_n$ which is a vector comprised of the delayed versions of the input signal x_n. Instead of computing the exact gradient, Widrow and fellow researchers [9,10] proposed the instantaneous gradient which only considered the most recent data samples (both input and desired). This led to the development of the stochastic gradient algorithm for MSE minimization that is popularly known as the Least Mean Squares (LMS) algorithm. The stochastic gradient is given by

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = -2e_k \mathbf{x}_k \tag{1.4}$$

Once the instantaneous gradient is known, the search should be in the direction opposite to the gradient which gives us the stochastic LMS algorithm in (1.5).

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \eta(k) e_k \mathbf{x}_k \tag{1.5}$$

The term $\eta(k)$ denotes a time-varying step-size that is typically chosen from a set of small positive numbers. Under mild conditions, it is possible to show that the LMS

algorithm converges in the mean to the Wiener solution [10-14]. The stochastic LMS algorithm is linear in complexity, i.e., $O(N)$, and allows on-line, local computations. These nice features facilitate efficient hardware implementation for real-world adaptive systems. Being a stochastic gradient algorithm, LMS suffers from problems related to slow convergence and excessive misadjustment in the presence of noise [14,15]. Higher order methods have been proposed to mitigate these effects and mainly they are variants of Quasi-Newton, Levenberg-Marquardt (LM) and Conjugate-Gradient (CG) methods popular in optimization [16-17]. Alternatively, we can derive a recursive fixed-point algorithm to iteratively estimate the optimal Wiener solution. This is the well-known Recursive Least Squares (RLS) algorithm [18,19].

Recursive Least Squares (RLS) Algorithm

The derivation of the RLS algorithm utilizes the fact that the input covariance matrix **R** can be iteratively estimated from its past values using the recursive relation,

$$\mathbf{R}(k) = \mathbf{R}(k-1) + \mathbf{x}_k \mathbf{x}_k^T \tag{1.6}$$

The above equation can also be viewed as a rank-1 update on the input covariance matrix **R**. Further, the cross correlation vector **P** satisfies the following recursion.

$$\mathbf{P}(k) = \mathbf{P}(k-1) + \mathbf{x}_k d_k \tag{1.7}$$

We know that the optimal Wiener solution at the time instant k is simply

$$\mathbf{w}_*(k) = \mathbf{R}^{-1}(k)\mathbf{P}(k) \tag{1.8}$$

Recall the matrix inversion lemma [7,8] at this point which allows us to recursively update the inverse of a matrix.

$$\mathbf{R}^{-1}(k) = \mathbf{R}^{-1}(k-1) - \frac{\mathbf{R}^{-1}(k-1)\mathbf{x}_k \mathbf{x}_k^T \mathbf{R}^{-1}(k-1)}{1 + \mathbf{x}_k^T \mathbf{R}^{-1}(k-1)\mathbf{x}_k} \tag{1.9}$$

It is important to note that the inversion lemma is useful only when the matrix itself can be expressed using reduced rank updates as in equation (1.6). By plugging equation (1.9) into the Wiener solution in (1.8) and using the recursive update for $\mathbf{P}(k)$ in (1.7), we can derive the RLS algorithm outlined in Table 1-1 below.

Table 1-1. Outline of the RLS Algorithm.

Initialize $\mathbf{R}^{-1}(0) = c\mathbf{I}$, where c is a large positive constant
$\mathbf{w}(0) = \mathbf{0}$, initialize the weight vector to an all zero vector
At every iteration, compute

$$\boldsymbol{\kappa}(k) = \frac{\mathbf{R}^{-1}(k-1)\mathbf{x}_k}{1+\mathbf{x}_k^T\mathbf{R}^{-1}(k-1)\mathbf{x}_k}$$

$$e(k) = d_k - \mathbf{w}^T(k-1)\mathbf{x}_k$$

$$\mathbf{w}(k) = \mathbf{w}(k-1) + \boldsymbol{\kappa}(k)\mathbf{e}(k)$$

$$\mathbf{R}^{-1}(k) = \mathbf{R}^{-1}(k-1) - \boldsymbol{\kappa}(k)\mathbf{x}_k^T\mathbf{R}^{-1}(k-1)$$

The RLS algorithm is a truly fixed-point method as it tracks the exact Wiener solution at every iteration. Also, observe the complexity of the algorithm is $O(\mathrm{N}^2)$ as compared to the linear complexity of the LMS algorithm. This additional increase in complexity is compensated by the fast convergence and zero misadjustment of the RLS algorithm.

Other Algorithms

Although LMS and RLS form the core of adaptive signal processing algorithms, researchers have proposed many other variants possessing varying degrees of complexity and performance levels. Important amongst them are the sign LMS algorithms that were introduced for reduced complexity hardware implementations [20, 21]. Historically, the sign-error algorithm has been utilized in the design of channel equalizers [20] and also in the 32kbps ADPCM digital coding scheme [22]. In terms of improving the speed of convergence with minimum misadjustment, variable step-size LMS and normalized LMS algorithms have been proposed [23-27]. Leaky LMS algorithms [28] have been explored

to mitigate the finite word length effects at the expense of introducing some bias in the optimal solution. Several extensions to the RLS algorithm have also been studied. Some of these algorithms show improved robustness against round-off errors and superior numerical stability [29,30]. The conventional RLS algorithm works well when the data statistics do not change over time (stationarity assumption). Analysis of the RLS tracking abilities in non-stationary conditions have been studied by Eleftheriou and Falconer [31] and many solutions have been proposed [14].

Limitations of MSE Criterion Based Linear Adaptive Systems

Although MSE based adaptive systems have been very popular, the criterion may not be the optimal choice for many engineering applications. For instance, consider the problem of system identification [32] which is stated as follows: Given a set of input and output noisy measurements where the outputs are the responses of an unknown system, obtain a parametric model estimate of the unknown system. If the unknown system is nonlinear, then it is obvious that MSE minimization would not result in the best possible representation of the system (plant). Criteria that utilize higher order statistics like the *error entropy*, for instance, can potentially provide a better model [33,34].

Let us restrict ourselves to the class of linear parametric models. Although the Wiener solution is optimal in the least squares sense, the biased input covariance matrix **R,** in the presence of additive white input noise yields a bias[1] in the optimal solution compared to what would have been obtained with noise-free data. This is a major drawback, since noise is omnipresent in practical scenarios. In order to illustrate the degradation in the quality of the parameter estimate, we created a random input time

[1] The Wiener solution with noise-free data gives *unbiased* estimates. We refer to this mismatch in the estimates obtained with and without noise as the *bias* introduced by noise.

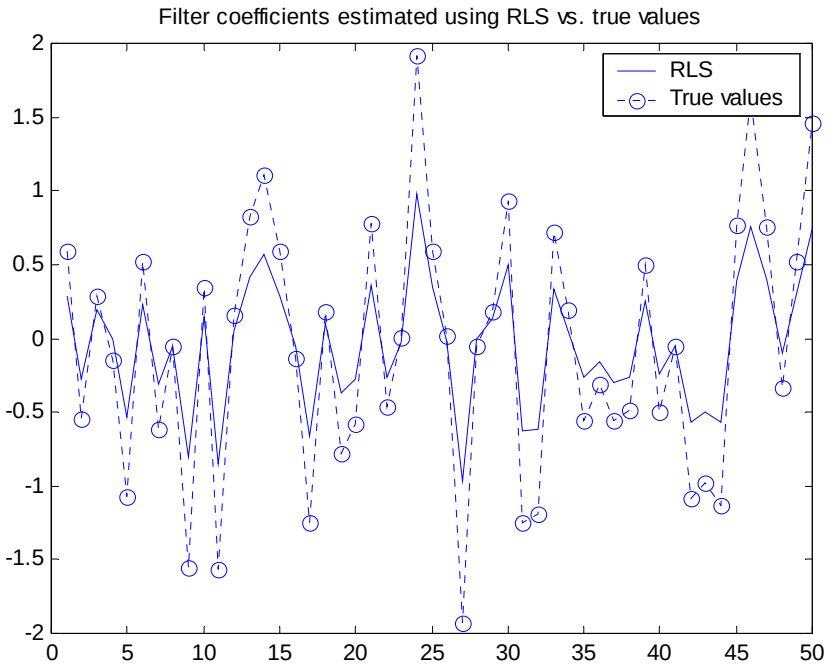


Figure 1-2. Parameter estimates using RLS algorithm with noisy data.

series with arbitrary coloring and passed it through a FIR filter with 50 taps. The filtered data were used as the desired signal. Uncorrelated white noise was added to the colored input signal and the input signal-to-noise ratio (SNR) was fixed at 0dB. The RLS algorithm was then used to estimate the weight vector. Ideally, if the SNR was infinite, RLS would have resulted in a weight vector exactly matching the FIR filter. However, because of the noisy input, the RLS estimates were biased as can be seen in Figure 1-2. This is a very serious drawback of the MSE criterion which is further accentuated by the fact that the *optimal Wiener MSE solution varies with changing noise power*. Researchers have dwelt on this problem for many years and several modifications have been proposed to mitigate the effect of noise on the estimate. Total least-squares (TLS) is one method which is quite powerful in eliminating the bias due to noise [35-42]. The instrumental variables (IV) method proposed as an extension to the Least-Squares (LS) has been previously applied for parameter estimation in white noise [32]. This method requires

choosing a set of *instruments* that are uncorrelated with the noise in the input [32,43]. Yet another classical approach is subspace Wiener filtering [14,44]. This approach tries to suppress the bias by performing an optimal subspace projection (Principal Component Space) and then training a filter in the reduced input space. In the next few sections, we will briefly cover some of these methods and discuss their benefits and the limitations.

Total Least Squares (TLS) and Other Methods

Mathematically speaking, TLS solves an over-determined set of linear equations of the form $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A} \in \Re^{m \times n}$ is the data matrix, $\mathbf{b} \in \Re^{m}$ is the desired vector, and $\mathbf{x} \in \Re^{n}$ is the parameter vector and *m* denotes the number of different observation vectors each of dimension *n* [41]. Alternatively, the linear equations can be written in the form $[\mathbf{A};\mathbf{b}][\mathbf{x}^T;-1] = \mathbf{0}$, where $[\mathbf{A};\mathbf{b}]$ denotes an augmented data matrix. Let **S** be the SVD [8] of the augmented data matrix $[\mathbf{A};\mathbf{b}]$ such that $\mathbf{S} = \mathbf{U\Sigma V}^T$, where $\mathbf{U}^T\mathbf{U} = \mathbf{I}_m$, $\mathbf{V}^T\mathbf{V} = \mathbf{I}_{n+1}$ and $\mathbf{\Sigma} = [diag(\sigma_1, \sigma_2, \sigma_3, \sigma_4, ..., \sigma_{n+1}); \mathbf{0}_{(m-n-1 \times n+1)}]$ with all singular values $\sigma_k > 0$. If $[\mathbf{A};\mathbf{b}][\mathbf{x}^T;-1] = \mathbf{0}$, the smallest singular value must be zero. This is possible only if $[\mathbf{x}^T;-1]$ is a singular vector of $[\mathbf{A};\mathbf{b}]$ (corresponding to the zero singular value) normalized such that its $(n+1)^{th}$ element value is -1. When $[\mathbf{A};\mathbf{b}]$ is a symmetric square matrix, the solution reduces to finding the eigenvector corresponding to the smallest eigenvalue of $[\mathbf{A};\mathbf{b}]$. The TLS solution in this special case is then

$$[\mathbf{x};-1] = -\mathbf{v}_{n+1} / \mathbf{v}_{n+1,n+1} \tag{1.10}$$

where $\mathbf{v}_{n+1,n+1}$ is the last element of the minor eigenvector $\mathbf{v}_{n+1}$. The Total Least-Squares technique can be easily applied to estimate the optimal solution using minor components estimation algorithms [45-51]. The computation of the TLS solution requires efficient

algorithms for extracting the principal components [52] or the eigenvectors of the data covariance matrix. Eigendecomposition is a well studied problem and many algorithms have been proposed for online estimation of eigenvectors and eigenvalues directly from data samples [53-77]. We have proposed robust, sample efficient algorithms for solving Principal Components Analysis (PCA) that have outperformed most of the available methods. A brief review of PCA theory and the proposed algorithms are outlined in appendix A. Brief mathematical analyses of the proposed algorithms according to the principles of stochastic approximation theory [78-85] are also included. A fast minor components analysis (MCA) based TLS algorithm [86] is discussed in appendix B.

Limitations of TLS

Total least squares gives unbiased estimates only when both the noise in the input and the desired data are independent and identically distributed (i.i.d.) and have same variance. Further, when the noise is truly i.i.d. Gaussian-distributed, the TLS solution is also the maximum likelihood solution. However, the assumption of equal noise variances is very restrictive, as measurement noises seldom have similar variances. The Generalized TLS (GTLS) problem [87] specifically deals with cases where the noise (still assumed to be i.i.d.) variances are different. However, the caveat is that the ratio of noise variances is assumed to be known which is, once again, not a practical assumption.

Extended TLS for Correlated Noise

In order to overcome the i.i.d. assumption, Mathews and Cichocki have proposed the Extended TLS (ETLS) [88] that allows the noise to have non-zero correlations. We will briefly describe the approach they adopted. Let the augmented input matrix [**A;b**] be

represented as $\overline{\mathbf{H}} = [\mathbf{A};\mathbf{b}]$. Then, the square matrix $\overline{\mathbf{H}}^T\overline{\mathbf{H}}$ can be written as a combination of the clean data matrix $\mathbf{H}^T\mathbf{H}$ and the noise covariance matrix $\mathbf{R}_N$.

$$\overline{\mathbf{H}}^T\overline{\mathbf{H}} = \mathbf{H}^T\mathbf{H} + \mathbf{R}_N \tag{1.11}$$

The above equation is true when the noise is uncorrelated with the clean data. This assumption is reasonable as the noise processes in general are unrelated (hence independent) to the physical sources that produced the data. Assume that there exists a matrix transformation $\widetilde{\mathbf{H}}$, such that

$$\widetilde{\mathbf{H}} = \overline{\mathbf{H}}\mathbf{R}_N^{-1/2} \tag{1.12}$$

The transformed data correlation matrix of $\widetilde{\mathbf{H}}$ is simply

$$\widetilde{\mathbf{H}}^T\widetilde{\mathbf{H}} = \mathbf{R}_N^{-1/2}\overline{\mathbf{H}}^T\overline{\mathbf{H}}\mathbf{R}_N^{-1/2} + \mathbf{I} \tag{1.13}$$

Equation (1.13) basically tells us that the transformed data are now corrupted by an i.i.d. noise process. Hence, we can now find the regular TLS solution with the transformed data by estimating the minor eigenvector of the matrix $\widetilde{\mathbf{H}}^T\widetilde{\mathbf{H}}$. In other words, the optimal ETLS solution for correlated noise signals is given by estimating the generalized eigenvector corresponding to the smallest generalized eigenvalue of the matrix pencil ($\overline{\mathbf{H}}^T\overline{\mathbf{H}}$, $\mathbf{R}_N$). Solving the generalized eigenvalue problem [8] is a non-trivial task and there are only a handful of algorithms that can provide online solutions. Our research in the area of PCA provided us the tools to develop a novel generalized eigenvalue decomposition (GED) algorithm. A short summary of the GED problem, existing learning algorithms and the proposed algorithm are listed in appendix C.

Although the ETLS seems to solve the general problem of linear parameter estimation, there is an inherent drawback. The ETLS requires the full knowledge of the

correlation matrix of the noise ($\mathbf{R}_N$). This assumption potentially leaves the problem of linear parameter estimation with noisy data wide open.

Other Methods

Infinite Impulse Response (IIR) system identification methods [89-92] deal with the problem of measurement noise in the output (desired) data. The Instrumental Variables (IV) method [93] for IIR system identification on the other hand, does not guarantee stability. It has been known for quite a while that the unit norm constraint for the equation-error (EE) based system identification is much better compared to the conventional monic constraint [90-92]. However, imposing the unit norm constraint appears too restrictive and hence limits the applicability.

Summary

In this chapter, we started by describing linear adaptive systems criteria and their associated algorithms. Most often, adaptive solutions are derived using the MSE criterion. We showed that the MSE criterion produces *biased* solutions in the presence of additive noise. The optimal Wiener MSE solution varies with changing noise variances which is highly undesirable. Alternative approaches to combat the effect of noise in the parameter estimation have been explored. The most popular approaches are based on the Total Least-Squares principles. Generalized TLS and Extended TLS improve upon the ability of the TLS to provide bias free estimates in the presence of additive noise. However, these methods rely on assumptions that can be very restrictive for real-world applications. Further, they require SVD and Generalized SVD computation [94-105] which increases the complexity. Another method called subspace Wiener filtering relies on the accurate estimation of the signal subspace from the noisy data. This technique

reduces the effect of the bias when the signals are distinguishable from noise (high SNR scenario). Otherwise, it fails since noise and signal subspaces cannot be separated.

Thus, it would not be fallacious to say that the problem of linear parameter estimation with noisy data is a hard problem that does not yet have a satisfactory solution in the existing literature. One of the major contributions of this dissertation is the development of an elegant solution to this problem without making any unreasonable assumptions about the noise statistics. Towards this end, we will present a new criterion based on the error signal and derive new learning algorithms.

CHAPTER 2
AUGMENTED ERROR CRITERION FOR LINEAR ADAPTIVE SYSTEMS

Introduction

In the previous chapter, we discussed the Mean-Squared Error (MSE) criterion which has been the workhorse of linear optimization theory due to the simple and analytically tractable structure of linear least squares. In adaptive filter theory, the classical Wiener-Hopf equations [6,10] are more commonly used owing to the extension of least squares to functional spaces (Hilbert spaces [106]) proposed by Wiener [6]. However, for finite impulse response (FIR) filters, (vector spaces) the two solutions coincide. There are also a number of important properties that help us understand the statistical properties of the Wiener solution, namely the orthogonality of the error signal to the input vector space as well as the whiteness of the predictor error signal for stationary inputs, provided the filter is long enough [5,14]. However, in a number of applications of practical importance, the error sequence produced by the Wiener filter is not white. One of the most important is the case of inputs corrupted by white noise, where the Wiener solution is biased by the noise variance as we saw before in Chapter 1.

In this chapter, we will develop a new criterion which augments the MSE criterion. In fact, MSE becomes a special case of this new criterion which we call the Augmented Error Criterion (AEC). Further, we will show that, under some conditions, this new criterion can produce a partially white error sequence at the output of an adaptive system even with noisy data. This special case of the AEC is called the Error Whitening Criterion (EWC). Our approach in this chapter will be as follows. We will first focus on

the problem of parameter estimation with noisy data and motivate the derivation of the error whitening criterion. Then, we will deduce the more generic augmented error criterion.

Error Whitening Criterion (EWC)

Consider the problem of parameter estimation with noisy data. Instead of minimizing the MSE, we will tackle the problem by introducing a new adaptation criterion that enforces zero autocorrelation of the error signal beyond a certain lag; hence the name *error whitening criterion* (EWC). Since we want to preserve the on-line properties of the adaptation algorithms, we propose to expand the error autocorrelation around a lag larger than the filter length using Taylor series. Thus, instead of an error signal, we will end up with an *error vector*, containing as many components as the terms kept in the Taylor series expansion. A schematic diagram of the proposed adaptation structure is depicted in Figure 2-1. The properties of this solution are very interesting, and it contains the Wiener solution as a special case. Additionally, for the case of two error terms, the same analytical tools developed for the Wiener filter can be applied with minor modifications. Moreover, when the input signal is contaminated with additive white

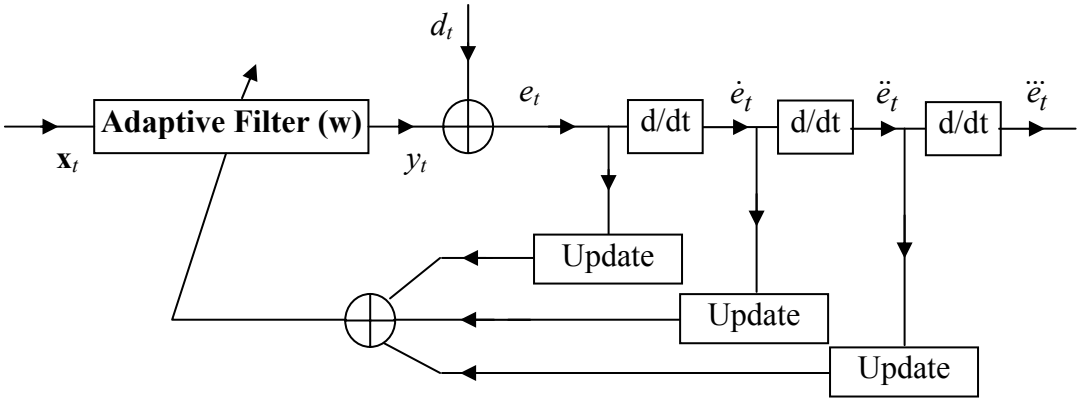


Figure 2-1. Schematic diagram of EWC adaptation.

noise, EWC produces the same optimal solution that would be obtained with the noise free data, with the same computational complexity of the Wiener solution.

Motivation for Error Whitening Criterion

The classical Wiener solution yields a biased estimate of the reference filter weight vector in the presence of input noise. This problem arises due to the contamination of the input signal autocorrelation matrix with that of the additive noise. If a signal is contaminated with additive white noise, only the zero-lag autocorrelation is biased by the amount of the noise power. Autocorrelation values at all other lags still remain at their original values. This observation rules out MSE as a good optimization criterion for this case. In fact, since the error power is the value of the error autocorrelation function at zero lag, the optimal weights will be biased because they depend on the input autocorrelation values at zero-lag. The fact that the autocorrelation values at non-zero lags are unaffected by the presence of noise will be proved useful in determining an unbiased estimate of the filter weights.

Analysis of the Autocorrelation of the Error Signal

The question that arises is what lag should be used to obtain the true weight vector in the presence of white input noise. Let us consider the autocorrelation of the training error at non-zero lags. Suppose noisy training data of the form $(\mathbf{x}(t), d(t))$ are provided, where $\mathbf{x}(t) = \tilde{\mathbf{x}}(t) + \mathbf{v}(t)$ and $d(t) = \tilde{d}(t) + u(t)$ with $\tilde{\mathbf{x}}(t)$ being the sample of the noise-free input vector at time t (time is assumed to be continuous), $\mathbf{v}(t)$ being the additive white noise vector on the input vector, $\tilde{d}(t)$ being the noise-free desired output and $u(t)$ being the additive white noise on the desired output. Suppose that the true weight vector of the reference filter that generated the data is $\mathbf{w}_T$ (moving average model). Then the

error at time t is $e(t) = (\widetilde{d}(t) + u(t)) - (\widetilde{\mathbf{x}}(t) + \mathbf{v}(t))^T \mathbf{w}$, where $\mathbf{w}$ is the estimated weight vector. Equivalently, when the desired response belongs to the subspace of the input, i.e., $\widetilde{d}(t) = \widetilde{\mathbf{x}}^T(t)\mathbf{w}_T$, the error can be written as

$$e(t) = (\widetilde{\mathbf{x}}^T(t)\mathbf{w}_T + u(t)) - (\widetilde{\mathbf{x}}(t) + \mathbf{v}(t))^T \mathbf{w} = \widetilde{\mathbf{x}}^T(t)(\mathbf{w}_T - \mathbf{w}) + u(t) - \mathbf{v}^T(t)\mathbf{w} \qquad (2.1)$$

Given this noisy training data, the MSE-based Wiener solution will not yield a residual training error that has zero autocorrelation for a number of consecutive lags, even when the contaminating noise signals are white. From (2.1) it is easy to see that the error will have a zero autocorrelation function if and only if

- the weight vector is equal to the true weights of the reference model,
- the lag is beyond the Wiener filter length.

During adaptation, the issue is that the filter weights are not set at $\mathbf{w}_T$, so the error autocorrelation function will be generally nonzero. Therefore a criterion to determine the true weight vector when the data is contaminated with white noise should be *to force the long lags (beyond the filter length) of the error autocorrelation function to zero by using an appropriate criterion*. This is exactly what the error-whitening criterion (EWC) that we propose here will do. There are two interesting situations that we should consider:

- What happens when the selected autocorrelation lag is smaller than the filter length?
- What happens when the selected autocorrelation lag is larger than the lag at which the autocorrelation function of the input signal vanishes?

The answer to the first question is simply that the solution will be still biased since it will be obtained by inverting a biased input autocorrelation matrix. If the selected lag is $L<m$ (m order of the reference filter), the bias will occur at the L^{th} sub-diagonal of the autocorrelation matrix, where the zero-lag autocorrelation of the input signal shows up. In

the special case of MSE, the selected lag is zero and the zeroth sub-diagonal becomes the main diagonal, thus the solution is biased by the noise power.

The answer to the second question is equally important. The MSE solution is quite stable because it is determined by the inverse of a diagonally dominant Toeplitz matrix. The diagonal dominance is guaranteed by the fact that the autocorrelation function of a real-valued function has a peak at zero-lag. If other lags are used in the criterion, it is important that the lag is selected such that the corresponding autocorrelation matrix (which will be inverted) is not ill conditioned. If the selected lag is larger than the length of the input autocorrelation function, then the autocorrelation matrix becomes singular and a solution cannot be obtained. Therefore, lags beyond the input signal correlation time should also be avoided in practice.

The observation that, constraining the higher lags of the error autocorrelation function to zero yields unbiased weight solutions is quite significant. Moreover, the algorithmic structure of this new solution and the lag-zero MSE solution are still very similar. The noise-free case helps us understand why this similarity occurs. Suppose the desired signal is generated by the following equation: $\tilde{d}(t) = \tilde{\mathbf{x}}^T(t)\mathbf{w}_T$, where $\mathbf{w}_T$ is the true weight vector. Now multiply both sides by $\tilde{\mathbf{x}}(t-\Delta)$ from the left and then take the expected value of both sides to yield $E[\tilde{\mathbf{x}}(t-\Delta)\tilde{d}(t)] = E[\tilde{\mathbf{x}}(t-\Delta)\tilde{\mathbf{x}}^T(t)]\mathbf{w}_T$. Similarly, we can obtain $E[\tilde{\mathbf{x}}(t)\tilde{d}(t-\Delta)] = E[\tilde{\mathbf{x}}(t)\tilde{\mathbf{x}}^T(t-\Delta)]\mathbf{w}_T$. Adding the corresponding sides of these two equations yields

$$E[\tilde{\mathbf{x}}(t)\tilde{d}(t-\Delta) + \tilde{\mathbf{x}}(t-\Delta)\tilde{d}(t)] = E[\tilde{\mathbf{x}}(t)\tilde{\mathbf{x}}^T(t-\Delta) + \tilde{\mathbf{x}}(t-\Delta)\tilde{\mathbf{x}}^T(t)]\mathbf{w}_T \tag{2.2}$$

This equation is similar to the standard Wiener-Hopf equation [9,10]

$E[\tilde{\mathbf{x}}(t)\tilde{d}(t)] = E[\tilde{\mathbf{x}}(t)\tilde{\mathbf{x}}^T(t)]\mathbf{w}_T$. Yet, it is different due to the correlations being evaluated at a lag other than zero, which means that the weight vector can be determined by constraining higher order lags in the error autocorrelation. Now that we have described the structure of the solution, let us address the issue of training linear systems using error correlations. Adaptation exploits the sensitivity of the error autocorrelation with respect to the weight vector of the adaptive filter. We will formulate the solution in continuous time first, for the sake of simplicity. If the support of the impulse response of the adaptive filter is of length *m*, we evaluate the derivative of the error autocorrelation function with respect to the lag Δ, where $\Delta \geq m$ are both real numbers. Assuming that the noises in the input and desired are uncorrelated to each other and the input signal, we get

$$\begin{aligned}
\frac{\partial \rho_e(\Delta)}{\partial \mathbf{w}} &= \frac{\partial E[e(t)e(t-\Delta)]}{\partial \mathbf{w}} \\
&= \frac{\partial E\left[(\tilde{\mathbf{x}}^T(t)(\mathbf{w}_T - \mathbf{w}) + u(t) - \mathbf{v}^T(t)\mathbf{w})(\tilde{\mathbf{x}}^T(t-\Delta)(\mathbf{w}_T - \mathbf{w}) + u(t-\Delta) - \mathbf{v}^T(t-\Delta)\mathbf{w})\right]}{\partial \mathbf{w}} \\
&= \frac{\partial E\left[(\mathbf{w}_T - \mathbf{w})^T \tilde{\mathbf{x}}(t)\tilde{\mathbf{x}}^T(t-\Delta)(\mathbf{w}_T - \mathbf{w}) + (u(t) - \mathbf{v}^T(t)\mathbf{w})(u(t-\Delta) - \mathbf{v}^T(t-\Delta)\mathbf{w})\right]}{\partial \mathbf{w}} \\
&= \frac{\partial (\mathbf{w}_T - \mathbf{w})^T E\left[\tilde{\mathbf{x}}(t)\tilde{\mathbf{x}}^T(t-\Delta)\right](\mathbf{w}_T - \mathbf{w})}{\partial \mathbf{w}} \\
&= -2E\left[\tilde{\mathbf{x}}(t)\tilde{\mathbf{x}}^T(t-\Delta)\right](\mathbf{w}_T - \mathbf{w})
\end{aligned} \tag{2.3}$$

The identity in equation (2.3) immediately tells us that the sensitivity of the error autocorrelation with respect to the weight vector becomes zero, i.e., $\partial \rho_e(\Delta)/\partial \mathbf{w} = \mathbf{0}$, if $(\mathbf{w}_T - \mathbf{w}) = \mathbf{0}$. This observation emphasizes the following important conclusion: when given training data that is generated by a linear filter, but contaminated with *white* noise, it is possible to derive simple adaptive algorithms that could determine the underlying filter weights without bias. Furthermore, if $(\mathbf{w}_T - \mathbf{w})$ is not in the null space of

$E[\tilde{\mathbf{x}}(t)\tilde{\mathbf{x}}^T(t-\Delta)]$, then only $(\mathbf{w}_T - \mathbf{w}) = \mathbf{0}$ makes $\rho_e(\Delta) = \mathbf{0}$ and $\partial\rho_e(\Delta)/\partial\mathbf{w} = \mathbf{0}$. But looking at (2.3), we conclude that a proper delay depends on the autocorrelation of the input signal that is, in general, unknown. Therefore, the selection of the delay Δ is important. One possibility is to evaluate the error autocorrelation function at different lags $\Delta \geq m$ and check for a non zero input autocorrelation function for that delay, which will be very time consuming and inappropriate for on-line algorithms.

Instead of searching for a good lag-Δ, consider the Taylor series approximation of the autocorrelation function around a *fixed lag-L*, where $L \geq m$,

$$\begin{aligned}\rho_e(\Delta) &\approx \rho_e(L) + \dot{\rho}_e(L)(\Delta - L) + \frac{1}{2}\ddot{\rho}_e(L)(\Delta - L)^2 + \ldots \\ &= E[e(t)e(t-L)] - E[e(t)\dot{e}(t-L)](\Delta - L) + \frac{1}{2}E[e(t)\ddot{e}(t-L)](\Delta - L)^2 + \ldots\end{aligned} \tag{2.4}$$

In (2.4), $\dot{e}(t)$ and $\ddot{e}(t)$ (see Figure 2-1) represent the derivatives of the error signal with respect to the time index. Notice that we do not take the Taylor series expansion around zero-lag for the reasons indicated above. Moreover, L should be less than the correlation time of the input, such that the Taylor expansion has a chance of being accurate. But since we bring more lags in the expansion, the choice of the lag becomes less critical than in (2.3). In principle, the more terms we keep in the Taylor expansion the more constraints we are imposing on the autocorrelation of the error in adaptation. Therefore, instead of finding the weight vector that makes the actual gradient in (2.3) zero, we find the weight vector that makes the derivative of the approximation in (2.4) with respect to the weight vector zero.

If the adaptive filter is operating in discrete time instead of continuous time, the differentiation with respect to time can be replaced by a first-order forward difference,

$\dot{e}(n) = e(n) - e(n-L)$. Higher order derivatives can also be approximated by their corresponding forward difference estimates, e.g., $\ddot{e}(n) = e(n) - 2e(n-L) + e(n-2L)$, etc. Although the forward difference normally uses two consecutive samples, for reasons that will become clear in the following sections of the chapter, we will utilize two samples separated by L samples in time. The first-order truncated Taylor series expansion for the error autocorrelation function for lag Δ evaluated at L becomes

$$\begin{aligned} \rho_e(\Delta) &\approx E[e(n)e(n-L)] - E[e(n)(e(n) - e(n-L))](\Delta - L) \\ &= -(\Delta - L)E[e^2(n)] + (1 + \Delta - L)E[e(n)e(n-L)] \end{aligned} \tag{2.5}$$

Analyzing (2.5) we remark another advantage of the Taylor series expansion because the familiar MSE is part of the expansion. Notice also that as one forces $\Delta \to L$, the MSE term will disappear and only the lag-L error autocorrelation will remain. On the other hand, as $\Delta \to L - 1$ only the MSE term will prevail in the autocorrelation function approximation. Introducing more terms in the Taylor expansion will bring in error autocorrelation constraints from lags iL.

Augmented Error Criterion (AEC)

We are now in a position to formulate the augmented error criterion (AEC). To the regular MSE term, we add another function $E(\dot{e}^2)$ to result in the augmented error criterion as shown in equation (2.6).

$$J(\mathbf{w}) = E[e^2(n)] + \beta E[\dot{e}^2(n)] \tag{2.6}$$

where β is a real scalar parameter. Equivalently, (2.6) can also be written as

$$J(\mathbf{w}) = (1 + 2\beta)E[e^2(n)] - 2\beta E[e(n)e(n-L)] \tag{2.7}$$

which has the same form as in (2.5). Notice that when $\beta = 0$ we recover the MSE in (2.6) and (2.7). Similarly, we would have to select $\Delta = L$ in order to make the first-order

expansion identical to the exact value of the error autocorrelation function. Substituting the identity $(1+2\beta) = -(\Delta - L)$, and using $\Delta = L$, we observe that $\beta = -1/2$ eliminates the MSE term from the criterion. Interestingly, this value will appear in a later discussion, when we optimize β in order to reduce the bias in the solution introduced by input noise. If β is positive, then minimizing the cost function $J(\mathbf{w})$ is equivalent to minimizing the MSE but with a constraint that the error signal must be smooth. Thus, the weight vector corresponding to the minimum $J(\mathbf{w})$ will result in a higher MSE than the Wiener solution.

The same criterion can also be obtained by considering performance functions of the form

$$\begin{aligned} J(\mathbf{w}) &= E\left[\left\| \begin{bmatrix} e(n) & \sqrt{\beta}\dot{e}(n) & \sqrt{\gamma}\ddot{e}(n) & \dots \end{bmatrix}^T \right\|_2^2\right] \\ &= E[e^2(n)] + \beta E[\dot{e}^2(n)] + \gamma E[\ddot{e}^2(n)] + \dots \end{aligned} \tag{2.8}$$

where the coefficients β, γ, etc. are assumed to be positive. Notice that (2.8) is the L2 norm of a vector of different objective functions. The components of this vector consist of $e(n)$, $\dot{e}(n)$, $\ddot{e}(n)$, etc. Due to the equivalence provided by the difference approximations for derivative, these terms constrain the error autocorrelation at lags iL as well as the error power as seen in (2.8).

In summary, the AEC defined by equation (2.6) can take many forms and hence results in different optimal solutions.

- If β is 0, then AEC exactly becomes the MSE criterion
- If β is -0.5, then AEC becomes the EWC which will result in an unbiased estimate of the parameters even in the presence of noise
- If β is positive and not equal to 0, then the cost function minimizes a combination of MSE with a smoothness constraint

In the following sections, we will further elaborate on the properties of AEC.

Properties of Augmented Error Criterion

Shape of the Performance Surface

Suppose that noise-free training data of the form $(\tilde{\mathbf{x}}(n), \tilde{d}(n))$, generated by a linear system with weight vector $\mathbf{w}_T$ through $\tilde{d}(n) = \tilde{\mathbf{x}}^T(n)\mathbf{w}_T$, is provided. Assume without loss of generality that the adaptive filter and the reference filter are of the same length. This is possible since it is possible to pad $\mathbf{w}_T$ with zeros if it is shorter than the adaptive filter. Therefore, the input vector $\tilde{\mathbf{x}}(n) \in \Re^m$, the weight vector $\mathbf{w}_T \in \Re^m$ and the desired output $\tilde{d}(n) \in \Re$. Equation (2.6) has a quadratic form and has a unique stationary point. If $\beta \geq 0$, then this stationary point is a minimum. Otherwise, the Hessian of (2.6) might have mixed-sign eigenvalues. We demonstrate this fact with sample performance surfaces obtained for 2-tap FIR filters using $\beta = -1/2$.

For three differently colored training data, we obtain the AEC performance surfaced shown in Figure 2-2. In each row, the MSE performance surface, the AEC cost contour plot, and the AEC performance surface are shown for the corresponding training data. The eigenvalue pairs of the Hessian matrix of (2.6) are (2.35,20.30), (-6.13,5.21), and (-4.08,-4.14), for these representative cases in Figure 2-2. Clearly, it is possible for (2.6) to have a stationary point that is a minimum, a saddle point, or a maximum and we start to see the differences brought about by the AEC.

The performance surface is a weighted sum of paraboloids, which will complicate gradient-based adaptation, but will not affect search algorithms utilizing curvature information. We will discuss more on the search techniques later in this Chapter and also in Chapter 4.

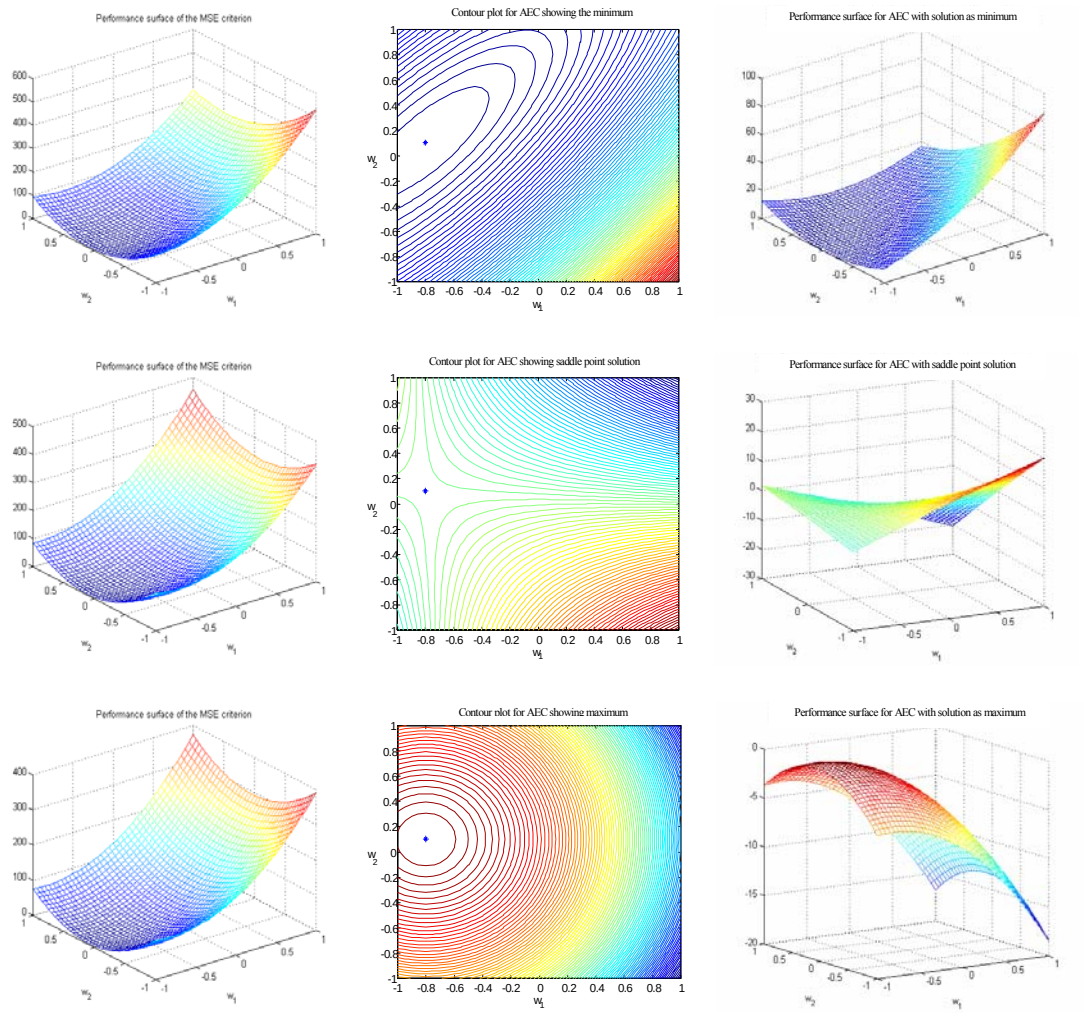


Figure 2-2. The MSE performance surfaces, the AEC contour plot, and the AEC performance surface for three different training data sets and 2-tap adaptive FIR filters.

Analysis of the Noise-free Input Case

Theorem 2.1: The stationary point of the quadratic form in (2.6) is given by

$$\mathbf{w}_* = (\widetilde{\mathbf{R}} + \beta\widetilde{\mathbf{S}})^{-1}(\widetilde{\mathbf{P}} + \beta\widetilde{\mathbf{Q}}) \tag{2.9}$$

where we defined $\widetilde{\mathbf{R}} = E[\widetilde{\mathbf{x}}(n)\widetilde{\mathbf{x}}^T(n)]$, $\widetilde{\mathbf{S}} = E[\dot{\widetilde{\mathbf{x}}}(n)\dot{\widetilde{\mathbf{x}}}^T(n)]$, $\widetilde{\mathbf{P}} = E[\widetilde{\mathbf{x}}(n)\widetilde{d}(n)]$ and $\widetilde{\mathbf{Q}} = E[\dot{\widetilde{\mathbf{x}}}(n)\dot{\widetilde{d}}(n)]$.

Proof: Substituting the proper variables in (2.6), we obtain the following explicit expression for $J(\mathbf{w})$.

$$J(\mathbf{w}) = E[\tilde{d}^2(n)] + \beta E[\dot{\tilde{d}}^2(n)] + \mathbf{w}^T(\tilde{\mathbf{R}} + \beta\tilde{\mathbf{S}})\mathbf{w} - 2(\tilde{\mathbf{P}} + \beta\tilde{\mathbf{Q}})^T\mathbf{w} \tag{2.10}$$

Taking the gradient with respect to $\mathbf{w}$ and equating to zero yields

$$\begin{aligned} &\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = 2(\tilde{\mathbf{R}} + \beta\tilde{\mathbf{S}})\mathbf{w} - 2(\tilde{\mathbf{P}} + \beta\tilde{\mathbf{Q}}) = \mathbf{0} \\ &\Rightarrow \mathbf{w}_* = (\tilde{\mathbf{R}} + \beta\tilde{\mathbf{S}})^{-1}(\tilde{\mathbf{P}} + \beta\tilde{\mathbf{Q}}) \end{aligned} \tag{2.11}$$

Notice that selecting $\beta = 0$ in (2.6) reduces the criterion to MSE and the optimal solution, given in (2.9), reduces to the Wiener solution. Thus, the Wiener filter is a special case of the AEC solution (though not optimal for noisy inputs, as we will show later).

Corollary 1. An equivalent expression for the stationary point of (2.6) is given by

$$\mathbf{w}_* = \left[(1+2\beta)\tilde{\mathbf{R}} - \beta\tilde{\mathbf{R}}_L\right]^{-1}\left[(1+2\beta)\tilde{\mathbf{P}} - \beta\tilde{\mathbf{P}}_L\right] \tag{2.12}$$

where we defined the matrix $\tilde{\mathbf{R}}_L = E[\tilde{\mathbf{x}}(n-L)\tilde{\mathbf{x}}^T(n) + \tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^T(n-L)]$ and the vector $\tilde{\mathbf{P}}_L = E[\tilde{\mathbf{x}}(n-L)\tilde{d}(n) + \tilde{\mathbf{x}}(n)\tilde{d}(n-L)]$. Notice that the interesting choice $\beta = -1/2$ yields $\mathbf{w}_* = \tilde{\mathbf{R}}_L^{-1}\tilde{\mathbf{P}}_L$.

Proof. Substituting the definitions of $\tilde{\mathbf{R}}$, $\tilde{\mathbf{S}}$, $\tilde{\mathbf{P}}$, $\tilde{\mathbf{Q}}$, and then recollecting terms to obtain $\tilde{\mathbf{R}}_L$ and $\tilde{\mathbf{P}}_L$ yields the desired result.

$$\begin{aligned} \mathbf{w}_* &= (\tilde{\mathbf{R}} + \beta\tilde{\mathbf{S}})^{-1}(\tilde{\mathbf{P}} + \beta\tilde{\mathbf{Q}}) \\ &= \left\{\begin{array}{c}\left[E[\tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^T(n)] + \beta E[(\tilde{\mathbf{x}}(n) - \tilde{\mathbf{x}}(n-L))(\tilde{\mathbf{x}}(n) - \tilde{\mathbf{x}}(n-L))^T]\right]^{-1} \\ \left[E[\tilde{\mathbf{x}}(n)\tilde{d}(n)] + \beta E[(\tilde{\mathbf{x}}(n) - \tilde{\mathbf{x}}(n-L))(\tilde{d}(n) - \tilde{d}(n-L))]\right]\end{array}\right\} \\ &= \left\{\begin{array}{c}\left[E[\tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^T(n)] + \beta(E[\tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^T(n)] + E[\tilde{\mathbf{x}}(n-L)\tilde{\mathbf{x}}^T(n-L)] - \tilde{\mathbf{R}}_L)\right]^{-1} \\ E[\tilde{\mathbf{x}}(n)\tilde{d}(n)] + \beta(E[\tilde{\mathbf{x}}(n)\tilde{d}(n)] + E[\tilde{\mathbf{x}}(n-L)\tilde{d}(n-L)] - \tilde{\mathbf{P}}_L)\end{array}\right\} \\ &= \left[(1+2\beta)\tilde{\mathbf{R}} - \beta\tilde{\mathbf{R}}_L\right]^{-1}\left[(1+2\beta)\tilde{\mathbf{P}} - \beta\tilde{\mathbf{P}}_L\right] \end{aligned} \tag{2.13}$$

From these results we deduct two extremely interesting conclusions:

Lemma 1. (Generalized Wiener-Hopf Equations) In the noise-free case, the true weight vector is given by $\widetilde{\mathbf{R}}_L \mathbf{w}_T = \widetilde{\mathbf{P}}_L$. (This result is also true for noisy data.)

Proof. This result follows immediately from the substitution of $\widetilde{d}(n) = \widetilde{\mathbf{x}}^T(n)\mathbf{w}_T$ and $\widetilde{d}(n-L) = \widetilde{\mathbf{x}}^T(n-L)\mathbf{w}_T$ in the definitions of $\widetilde{\mathbf{R}}_L$ and $\widetilde{\mathbf{P}}_L$.

Lemma 2. In the noise-free case, regardless of the specific value of β, the optimal solution is equal to the true weight vector, i.e., $\mathbf{w}_* = \mathbf{w}_T$.

Proof. This result follows immediately from the substitution of the result in *Lemma* 1 into the optimal solution expression given in (2.9).

The result in *Lemma* 1 is especially significant, since it provides a generalization of the Wiener-Hopf equations to autocorrelation and cross correlation matrices evaluated at different lags of the signals. In these equations, *L* represents the specific correlation lag selected, and the choice *L*=0 corresponds to the traditional Wiener-Hopf equations. The generalized Wiener-Hopf equations are essentially stating that, the true weight vector can be determined by exploiting correlations evaluated at different lags of the signals, and we are not restricted to the zero-lag correlations as in the Wiener solution.

Analysis of the Noisy Input Case

Now, suppose that we are given noisy training data $(\mathbf{x}(n), d(n))$, where $\mathbf{x}(n) = \widetilde{\mathbf{x}}(n) + \mathbf{v}(n)$ and $d(n) = \widetilde{d}(n) + u(n)$. The additive noise on both signals are zero-mean and uncorrelated with each other and with the input and desired signals. Assume that the additive noise, $u(n)$, on the desired is white (in time) and let the autocorrelation matrices of $\mathbf{v}(n)$ be $\mathbf{V} = E[\mathbf{v}(n)\mathbf{v}^T(n)]$, and $\mathbf{V}_L = E[\mathbf{v}(n-L)\mathbf{v}^T(n) + \mathbf{v}(n)\mathbf{v}^T(n-L)]$.

Under these circumstances, we have to estimate the necessary matrices to evaluate (2.9) using noisy data. These matrices evaluated using noisy data, $\mathbf{R}$, $\mathbf{S}$, $\mathbf{P}$, and $\mathbf{Q}$ will become (see appendix D for details)

$$\begin{aligned}
\mathbf{R} &= E[\mathbf{x}(n)\mathbf{x}^T(n)] = \widetilde{\mathbf{R}} + \mathbf{V} \\
\mathbf{S} &= E[(\mathbf{x}(n) - \mathbf{x}(n-L))(\mathbf{x}(n) - \mathbf{x}(n-L))^T] = 2(\widetilde{\mathbf{R}} + \mathbf{V}) - \widetilde{\mathbf{R}}_L - \mathbf{V}_L \\
\mathbf{P} &= E[\mathbf{x}(n)d(n)] = \widetilde{\mathbf{P}} \\
\mathbf{Q} &= E[(\mathbf{x}(n) - \mathbf{x}(n-L))(d(n) - d(n-L))^T] = 2\widetilde{\mathbf{P}} - \widetilde{\mathbf{P}}_L
\end{aligned} \tag{2.14}$$

Finally, the optimal solution estimate of AEC, when presented with noisy input and desired output data, will be

$$\begin{aligned}
\hat{\mathbf{w}}_* &= (\mathbf{R} + \beta\mathbf{S})^{-1}(\mathbf{P} + \beta\mathbf{Q}) \\
&= \left[(\widetilde{\mathbf{R}} + \mathbf{V}) + \beta(2(\widetilde{\mathbf{R}} + \mathbf{V}) - \widetilde{\mathbf{R}}_L - \mathbf{V}_L)\right]^{-1}\left[\widetilde{\mathbf{P}} + \beta(2\widetilde{\mathbf{P}} - \widetilde{\mathbf{P}}_L)\right] \\
&= \left[(1+2\beta)(\widetilde{\mathbf{R}} + \mathbf{V}) - \beta\widetilde{\mathbf{R}}_L - \beta\mathbf{V}_L\right]^{-1}\left[(1+2\beta)\widetilde{\mathbf{P}} - \beta\widetilde{\mathbf{P}}_L\right]
\end{aligned} \tag{2.15}$$

Theorem 2.2: (EWC Noise-Rejection Theorem) In the noisy-input data case, the optimal solution obtained using AEC will be identically equal to the true weight vector if and only if $\beta = -1/2$, $\widetilde{\mathbf{R}}_L \neq \mathbf{0}$, and $\mathbf{V}_L = \mathbf{0}$. There are two situations to consider:

- When the adaptive linear system is an FIR filter, the input noise vector v_k consists of delayed versions of a single dimensional noise process. In that case, $\mathbf{V}_L = \mathbf{0}$ if and only if $L \geq m$, where m is the filter length and the single dimensional noise process is white.
- When the adaptive linear system is an ADALINE, the input noise is a vector process. In that case, $\mathbf{V}_L = \mathbf{0}$ if and only if the input noise vector process is white (in time) and $L \geq 1$. The input noise vector may be spatially correlated.

Proof: Sufficiency of the first statement is immediately observed by substituting the provided values of β and $\mathbf{V}_L$. Necessity is obtained by equating (2.15) to $\mathbf{w}_T$ and substituting the generalized Wiener-Hopf equations provided in *Lemma* 1. Clearly, if $\widetilde{\mathbf{R}}_L = \mathbf{0}$, then there is no equation to solve, thus the weights cannot be uniquely

determined using this value of L. The statement regarding the FIR filter case is easily proved by noticing that the temporal correlations in the noise vector diminish once the autocorrelation lag becomes greater than equal to the filter length. The statement regarding the ADALINE structure is immediately obtained from the definition of a temporally white vector process.

Orthogonality of Error to Input

An important question regarding the behavior of the optimal solution obtained using the AEC is the relationship between the residual error signal and the input vector. In the case of MSE, we know that the Wiener solution results in the error to be orthogonal to the input signal, i.e., $E[e(n)\mathbf{x}(n)] = \mathbf{0}$ [10,14,15]. However, this result is true only when there is no noise and also when the estimated filter length is greater than the actual system impulse response. Similarly, we can determine what the AEC will achieve.

Lemma 3: At the optimal solution of AEC, the error and the input random processes satisfy $\beta E[e(n)\mathbf{x}(n-L) + e(n-L)\mathbf{x}(n)] = (1+2\beta)E[e(n)\mathbf{x}(n)]$, for all $L \geq 0$.

Proof: We know that the optimal solution of AEC for any $L \geq 0$ is obtained when the gradient of the cost function with respect to the weights is zero. Therefore,

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{w}} &= 2E[e(n)\mathbf{x}(n)] + 2\beta E[(e(n)-e(n-L))(\mathbf{x}(n)-\mathbf{x}(n-L))] \\ &= (1+2\beta)E[e(n)\mathbf{x}(n)] - \beta E[e(n)\mathbf{x}(n-L) + e(n-L)\mathbf{x}(n)] = \mathbf{0}\end{aligned} \tag{2.16}$$

It is interesting to note that if $\beta = -1/2$, then we obtain $E[e(n)\mathbf{x}(n-L) + e(n-L)\mathbf{x}(n)] = \mathbf{0}$ for all L. On the other hand, since the criterion reduces to MSE for $\beta = 0$, then we obtain $E[e(n)\mathbf{x}(n)] = \mathbf{0}$. The result shown in (2.16), if interpreted in terms of Newtonian physics, reveals an interesting insight as to the

behavior of the EWC criterion ($\beta = -1/2$) at its optimal solution (regardless of the length of the reference filter that created the desired signal). In a simplistic manner, this behavior could be summarized by the following statement: The optimal solution of EWC tries to decorrelate the residual error from the estimated future value of the input vector (see appendix E for details).

The case where $\beta = -1/2$ is especially interesting, because it results in complete noise rejection. Notice that, in this case, since the optimal solution is equal to the true weight vector, the residual error is given by $e(n) = u(n) - \mathbf{v}^T(n)\mathbf{w}_T$, which is composed purely of the noise in the training data. Certainly, this is the only way that the adaptive filter can achieve $E[e(n)\mathbf{x}(n-L) + e(n-L)\mathbf{x}(n)] = \mathbf{0}$ for all L values, since $E[e(n)\mathbf{x}(n-L)] = E[e(n-L)\mathbf{x}(n)] = \mathbf{0}$ for this error signal. Thus, EWC not only orthogonalizes the instantaneous error and input signals, but it orthogonalizes all lags of the error from the input.

Relationship to Error Entropy Maximization

Another interesting property that the AEC solution exhibits is its relationship with entropy [107]. Notice that when $\beta < 0$, the optimization rule tries to minimize MSE, yet it tries to maximize the separation between samples of errors, simultaneously. We could regard the sample separation as an estimate of the error entropy. In fact, the entropy estimation literature is full of methods based on sample separations [108-113]. Specifically, the EWC case with $\beta = -1/2$, finds the perfect balance between entropy and MSE that allows us to eliminate the effect of noise on the solution. Recall that the Gaussian density displays maximum entropy among distributions of fixed variance [114]. In the light of this fact, the aim of EWC could be understood as finding the minimum

error variance solution, while keeping the error close to Gaussian. Notice that, due to central limit theorem [114], the error signal will be closely approximated by a Gaussian density when there are a large number of taps. A brief description of the relationship between entropy (using estimators) [115-117] and sample differences is provided in appendix F.

Note on Model-Order Selection

Model order selection is another important issue in adaptive filter theory. The actual desired behavior from an adaptive filter is to find the right balance between approximating the training data as accurately as possible and generalizing to *unseen* data with precision [118]. One major cause of poor generalization is known to be excessive model complexity [118]. Under these circumstances, the designer's aim is to determine the least complex adaptive system (which translates to smaller number of weights in the case of linear systems) that minimizes the approximation error. Akaike's information criterion (AIC) [119] and Rissanen's minimum description length (MDL) [120] are two important theoretical results regarding model order selection. Such methods require the designer to evaluate an objective function, which is a combination of MSE and the filter length or the filter weights, using different lengths of adaptive filters.

Consider the case of overmodeling in the problem of linear FIR filter (assume N taps) estimation. If we use the MSE criterion, and assume that there is no noise in the data, then, the estimated Wiener solution will have exactly N non-zero elements that exactly match with the true FIR filter. This is a very nice property of the MSE criterion. However, when there is noise in the data, then this property of MSE is no longer true. Therefore, increasing the length of the adaptive filter will only result in more parameter bias in the Wiener solution. On the other hand, EWC successfully determines the length

of the true filter, even in the presence of additive noise. In the overmodeling case, the additional taps will decay to zero indicating that a smaller filter is sufficient to model the data. This is exactly what we would like an automated regularization algorithm to achieve: determining the proper length of the filter without requiring external discrete modifications on this parameter. Therefore, EWC extends the regularization capability of MSE to the case of noisy training data. Alternatively, EWC could be used as a criterion for determining the model order in a fashion similar to standard model order selection methods. Given a set of training samples, one could start solving for the optimal EWC solution for various lengths of the adaptive filter. As the length of the adaptive filter is increased past the length of the true filter, the error power with the EWC solution will become constant. Observing this point of transition from variable to constant error power, we can determine the exact model order of the original filter.

The Effect of β on the Weight Error Vector

The effect of the cost function free parameter β on the accuracy of the solution (compared to the true weight vector that generated the training data) is another crucial issue. In fact, it is possible to determine the dynamics of the weight error as a function of β. This result is provided in the following lemma.

Lemma 4: (The effect of β on AEC solution) In the noisy training data case, the derivative of the error vector between the optimal EWC solution and the true weight vector, i.e., $\hat{\boldsymbol{\varepsilon}}_* = \hat{\mathbf{w}}_* - \mathbf{w}_T$, with respect to β is given by

$$\frac{\partial \hat{\boldsymbol{\varepsilon}}_*}{\partial \beta} = -\left[(1+2\beta)(\mathbf{R}+\mathbf{V}) - \beta \mathbf{R}_L\right]^{-1}\left[2(\mathbf{R}-\mathbf{R}_L)\hat{\boldsymbol{\varepsilon}}_* - \mathbf{R}_L \mathbf{w}_T\right] \tag{2.17}$$

Proof: Recall from (2.15) that in the noisy data case, the optimal AEC solution is given

by $\hat{\mathbf{w}}_* = \left[(1+2\beta)(\mathbf{R}+\mathbf{V}) - \beta\mathbf{R}_L - \beta\mathbf{V}_L\right]^{-1}\left[(1+2\beta)\mathbf{P} - \beta\mathbf{P}_L\right]$. Using the chain rule for the derivative and the fact that for any nonsingular matrix $\mathbf{A}(\beta)$, $\partial\mathbf{A}^{-1}/\partial\beta = -\mathbf{A}^{-1}(\partial\mathbf{A}/\partial\beta)\mathbf{A}^{-1}$, the result in (2.17) follows from straightforward derivation. In order to get the derivative as $\beta \to -1/2$, we substitute this value and $\hat{\boldsymbol{\varepsilon}}_* = \mathbf{0}$.

The significance of *Lemma* 4 is that it shows that no finite β value will make this error derivative zero. The matrix inversion, on the other hand, approaches to zero for unboundedly growing β. In addition, it could be used to determine the Euclidean error norm derivative, $\partial\left\|\hat{\boldsymbol{\varepsilon}}_*\right\|_2^2/\partial\beta$.

Numerical Case Studies of AEC with the Theoretical Solution

In the preceding sections, we have built the theory of the augmented error criterion and its special case, the error whitening criterion, for linear adaptive filter optimization. We have investigated the behavior of the optimal solution as a function of the cost function parameters as well as determining the optimal value of this parameter in the noisy training data case. This section is designed to demonstrate these theoretical results in numerical case studies with Monte Carlo simulations. In these simulations, the following scheme will be used to generate the required autocorrelation and crosscorrelation matrices.

Given the scheme depicted in Figure 2-3, it is possible to determine the *true* analytic auto/cross-correlations of all signals of interest, in terms of the filter coefficients and the noise powers. Suppose ξ, $\widetilde{v}$, and u are zero-mean white noise signals with powers σ_x^2, σ_v^2, and σ_u^2, respectively. Suppose that the coloring filter **h** and the

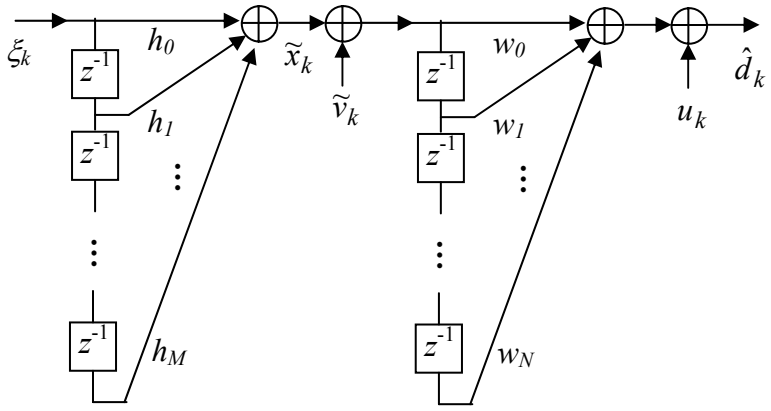


Figure 2-3. Demonstration scheme with coloring filter **h**, true mapping filter **w**, and the uncorrelated white signals.

mapping filter **w** are unit-norm. Under these conditions, we obtain

$$E[\tilde{x}(n)\tilde{x}(n-\Delta)] = \sigma_x^2 \sum_{j=0}^{M} h_j h_{j+\Delta} \tag{2.18}$$

$$E[(\tilde{x}(n)+\tilde{v}(n))(\tilde{x}(n-\Delta)+\tilde{v}(n-\Delta))] = \begin{cases} \sigma_x^2 + \sigma_v^2, & \Delta = 0 \\ E[\tilde{x}(n)\tilde{x}(n-\Delta)], & \Delta \neq 0 \end{cases} \tag{2.19}$$

$$E[(\tilde{x}(n)+\tilde{v}(n))\hat{d}(n)] = \sigma_v^2 w_{-\Delta} + \sum_{l=0}^{N} w_l E[\tilde{x}(n)\tilde{x}(n-l-\Delta)] \tag{2.20}$$

For each combination of SNR from {-10dB,0dB,10dB}, β from {-0.5,-0.3,0,0.1}, m from {2,…,10}, and L from {m,…,20} we have performed 100 Monte Carlo simulations using randomly selected 30-tap FIR coloring and n-tap mapping filters. The length of the mapping filters and that of the adaptive filters were selected to be equal in every case. In all simulations, we used an input signal power of $\sigma_x^2 = 1$, and the noise powers $\sigma_v^2 = \sigma_u^2$ are determined from the given SNR using $SNR = 10\log_{10}(\sigma_x^2/\sigma_v^2)$. The matrices $\mathbf{R}$, $\mathbf{S}$, $\mathbf{P}$, and $\mathbf{Q}$, which are necessary to evaluate the optimal solution given by (2.15) are then evaluated using (2.18), (2.19), and (2.20), analytically. The results obtained are summarized in Figure 2-4 and Figure 2-5, where for the three SNR levels selected, the

average squared error norm for the optimal solutions (in reference to the true weights) are given as a function of L and n for different β values. In Figure 2-4, we present the average normalized weight vector error norm obtained using AEC at different SNR levels and using different β values as a function of the correlation lag L that is used in the criterion. The filter length was fixed to 10 in these simulations.

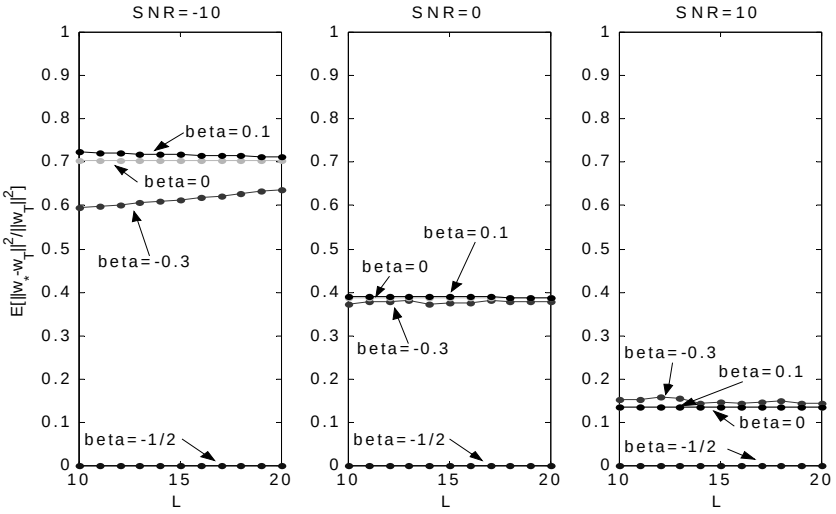


Figure 2-4. The average squared error-norm of the optimal weight vector as a function of autocorrelation lag L for various β values and SNR levels.

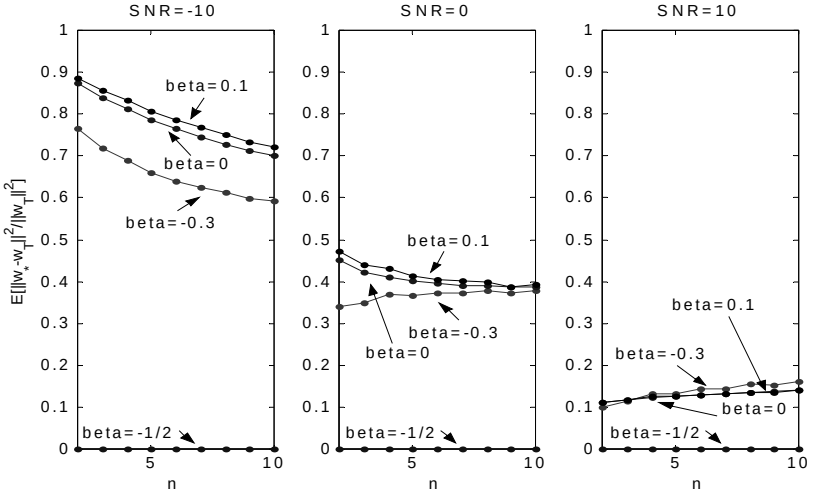


Figure 2-5. The average squared error-norm of the optimal weight vector as a function of filter length m for various β values and SNR levels.

From the theoretical analysis, we know that if the input autocorrelation matrix is invertible, then the solution accuracy should be independent of the autocorrelation lag L. The results of the Monte Carlo simulations presented in Figure 2-4 conform to this fact. As expected, the optimal choice of $\beta = -1/2$ determined the correct filter weights exactly. Another set of results, presented in Figure 2-5, shows the effect of filter length on the accuracy of the solutions provided by the AEC. The optimal value of $\beta = -1/2$ always yields the perfect solution, whereas the accuracy of the optimal weights degrades as this parameter is increased towards zero (i.e. as the weights approach the Wiener solution). An interesting observation from Figure 2-5 is that for SNR levels below zero, the accuracy of the solutions using sub-optimal β values increases, whereas for SNR levels above zero, the accuracy decreases when the filter length is increased. For zero SNR, on the other hand, the accuracy seems to be roughly unaffected by the filter length.

The Monte Carlo simulations performed in the preceding examples utilized the exact coloring filter and the true filter coefficients to obtain the analytical solutions. In our final case study, we demonstrate the performance of the batch solution of the AEC criterion obtained from sample estimates of all the relevant auto- and cross-correlation matrices. In these Monte Carlo simulations, we utilize 10,000 samples corrupted with white noise at various SNR levels. The results of these Monte Carlo simulations are summarized in the histograms shown in Figure 2-6. Each subplot of Figure 2-6 corresponds to experiments performed using SNR levels of –10 dB, 0dB, and 10 dB for each column and adaptive filter lengths of 4-taps, 8-taps, and 12-taps for each row, respectively. For each combination of SNR and filter length, we have performed 50 Monte Carlo simulations using MSE ($\beta = 0$) and EWC ($\beta = -1/2$) criteria. The

correlation lag is selected to be equal to the filter length in all simulations, due to *Theorem* 2.2. Clearly, Figure 2-6 demonstrates the superiority of the AEC in rejecting noise that is present in the training data. Notice that in all subplots (for all combinations of filter length and SNR), AEC achieves a smaller average error norm than MSE.

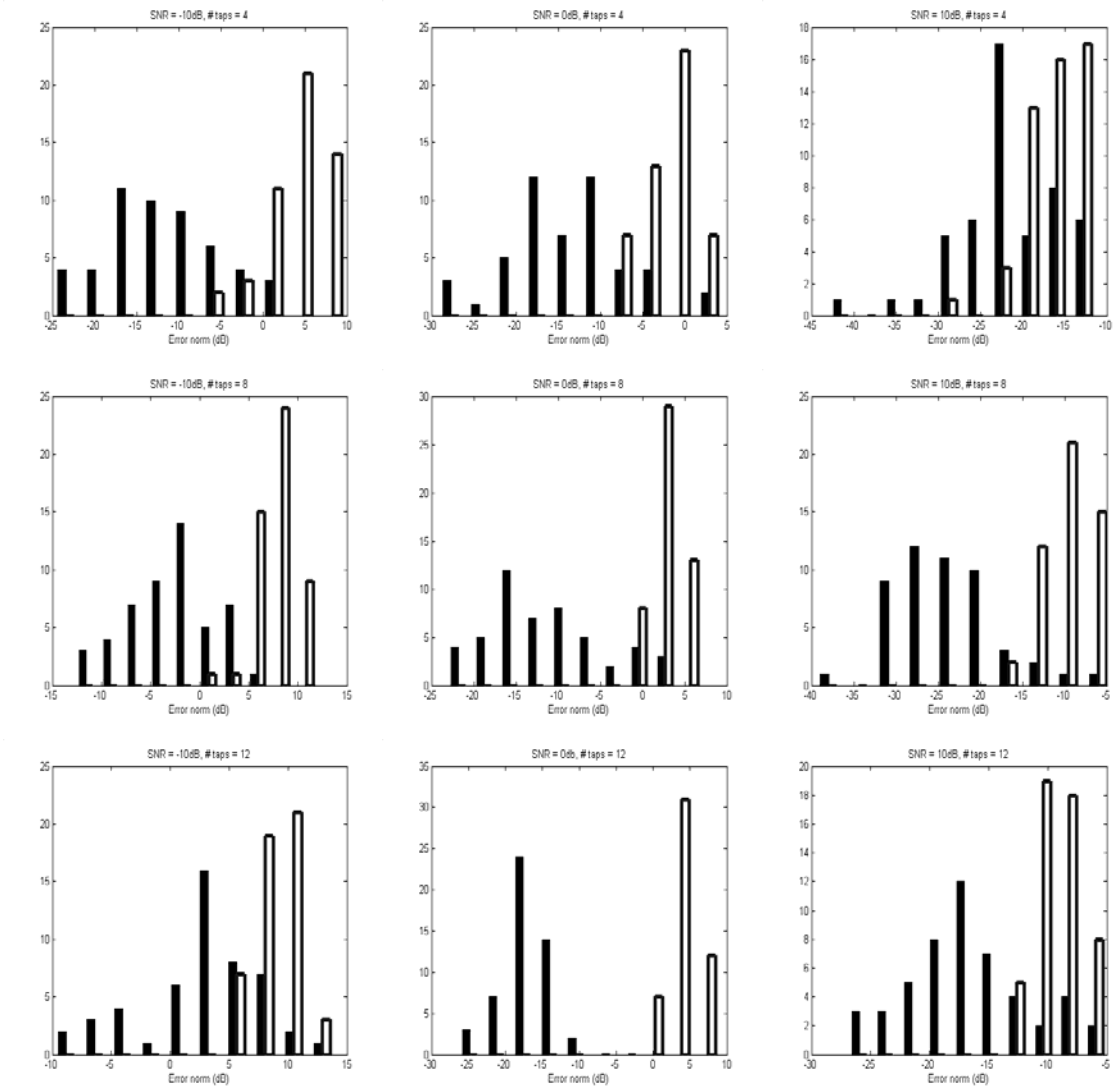


Figure 2-6. Histograms of the weight error norms (dB) obtained in 50 Monte Carlo simulations using 10000 samples of noisy data using MSE (empty bars) and EWC with β = -0.5 (filled bars). The subfigures in each row use filters with 4, 8, and 12 taps respectively. The subfigures in each column use noisy samples at –10, 0, and 10 dB SNR, respectively.

The discrepancy between the performances of the two solutions intensifies with increasing filter length. Next, we will demonstrate the error-whitening property of the EWC solution. From equation (2.1) we can expect that the error autocorrelation function will vanish at lags greater than or equal to the length of the reference filter, if the weight vector is identical to the true weight vector. For any other value of the weight vector, the error autocorrelation fluctuates at non-zero values. A 4-tap reference filter is identified with a 4-tap adaptive filter using noisy training data (hypothetical) at an SNR level of 0dB. The autocorrelation functions of the error signals corresponding to the MSE solution and the EWC solution are shown in Figure 2-7. Clearly, the EWC criterion determines a solution that forces the error autocorrelation function to zero at lags greater than or equal the filter length (partial whitening of the error).

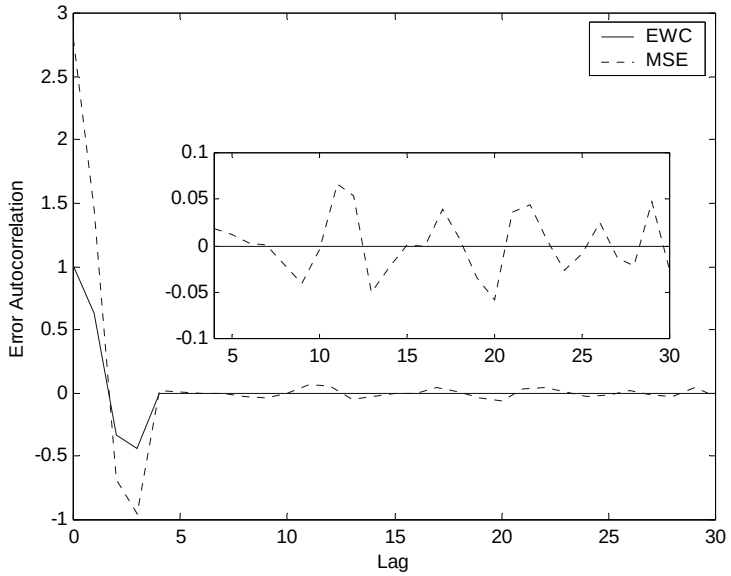


Figure 2-7. Error autocorrelation function for MSE (dotted) and EWC (solid) solutions.

Finally, we will address the order selection capability and demonstrate how the AEC (specifically EWC) can be used as a tool for determining the correct filter order, even with noisy data, provided that the given input-desired output pair is a moving

average process. For this purpose, we determine the theoretical Wiener and EWC (with $\beta = -1/2$ and $L = m$, where m is the length of the adaptive filter) solutions for a randomly selected pair of coloring filter, **h**, and mapping filter **w**, at different adaptive filter lengths. The noise level is selected to be 20 dB, and the length of the true mapping filter is 5. We know from our theoretical analysis that if the adaptive filter is longer than the reference filter, the EWC will yield the true weight vector padded with zeros. This will not change the MSE of the solution. Thus, if we plot the MSE of the EWC versus the length of the adaptive filter, starting from the length of the actual filter, the MSE curve will remain flat, whereas the Wiener solution will keep decreasing the MSE, contaminating the solution by learning the noise in the data. Figure 2-8(a) shows the MSE obtained with the Wiener solution as well as the EWC solution for different lengths of the adaptive filter using the same training data described above. Notice (in the zoomed-in portion) that the MSE with EWC remains constant starting from 5, which is the filter order that generated the data. On the other hand, if we were to decide on the filter order looking at the MSE of the Wiener solution, we would select a model order of 4, since the gain in MSE is insignificantly small compared to the previous steps from this point on. Figure 2-8(b) shows the norm of the weight vector error for the solutions obtained using the EWC and MSE criteria, which confirms that the true weight vector is indeed attained with the EWC criterion once the proper model order is reached.

This section aimed at experimentally demonstrating the theoretical concepts set forth in the preceding sections of the chapter. We have demonstrated with numerous Monte Carlo simulations that the analytical solution of the EWC criterion eliminates the effect of noise completely if the proper value is used for β. We have also demonstrated

that the batch solution of EWC (estimated from a finite number of samples) outperforms MSE in the presence of noise, provided that a sufficient number of samples are given so that the noise autocorrelation matrices diminish as required by the theory.

Summary

In this chapter, we derived the augmented error criterion (AEC) and discussed a special case of AEC called the error whitening criterion (EWC). The proposed AEC includes MSE as a special case. We discussed some of the interesting properties of the AEC cost function and worked out the analytical optimal solution. Further, we discussed the reasoning behind naming the special case of AEC with the parameter $\beta = -0.5$ as EWC. The intuitive reasoning is that this criterion partially whitens the error signal even in the presence of noise which cannot be achieved by the MSE criterion. Thus the error whitening criterion is very useful for estimating the parameters of a linear unknown system in the presence of additive white noise. AEC with other values of β can be used as a constrained MSE criterion where the constraint is the smoothness of the error signal. Most of the material presented in this chapter can be found in [121].

Although we have presented a complete theoretical investigation of the proposed criterion and its analytical solution, in practice, on-line algorithms that operate on a sample-by-sample basis to determine the desired solution are equally valuable. Therefore, in the following chapters, we will focus on designing computationally efficient on-line algorithms to solve for the optimal AEC solution in a fashion similar to the well-known RLS and LMS algorithms. In fact, we aim to come up with algorithms that have the same computational complexity with these two widely used algorithms.

CHAPTER 3
FAST RECURSIVE NEWTON TYPE ALGORITHMS FOR AEC

Introduction

In Chapter 2, we derived the analytical solution for AEC. We also showed simulation results using block methods. In this Chapter, the focus will be on deriving online, sample-by-sample Newton type algorithms to estimate the optimal AEC solution. First, we will derive a Newton type algorithm that has a structure similar to the well-known RLS algorithm that estimates the optimal Wiener solution for MSE criterion. The complexity of the proposed algorithm is $O(N^2)$ which is again comparable with that of the RLS algorithm. Then, we will propose another Newton type algorithm derived from the principles of TLS using minor components analysis. This algorithm in its current form estimates the optimal EWC solution which is a special case of AEC with β = -0.5.

Derivation of the Newton Type Recursive Error Whitening Algorithm

Given the estimate of the filter tap weights at time instant $(n-1)$, the goal is to determine the best set of tap weights at the next iteration n that would track the optimal solution. We call this algorithm as Recursive Error Whitening (REW) algorithm although the *error whitening* property is applicable only when the parameter β is set to -0.5. But, the algorithm can be applied with any value of β. Recall that the RLS algorithm belongs to the class of fixed-point algorithms in the sense that they track the optimal Wiener solution at every time step. The REW algorithm falls in the same category and it tracks the optimal AEC solution at every iteration. The noteworthy feature of the fixed-point algorithms is their exponential convergence rate as they utilize higher order information

like curvature of the performance surface. Although the complexity of the fixed-point Newton type algorithms is higher when compared to the conventional gradient methods, the superior convergence and robustness to the eigenspread of the data can be vital gains in many applications.

For convenience purposes, we will drop the *tilde* convention that we used in the previous chapter to differentiate between noise-corrupted and noise free matrices and vectors. Recall that the optimal AEC solution is given by

$$\mathbf{w}_* = (\mathbf{R} + \beta\mathbf{S})^{-1}(\mathbf{P} + \beta\mathbf{Q}) \tag{3.1}$$

Letting $\mathbf{T}(n) = \mathbf{R}(n) + \beta\mathbf{S}(n)$ and $\mathbf{V}(n) = \mathbf{P}(n) + \beta\mathbf{Q}(n)$, we obtain the following recursion.

$$\begin{aligned}\mathbf{T}(n) &= \mathbf{T}(n-1) + (1+2\beta)\mathbf{x}(n)\mathbf{x}^T(n) - \beta\mathbf{x}(n-L)\mathbf{x}^T(n) - \beta\mathbf{x}(n)\mathbf{x}^T(n-L) \\ &= \mathbf{T}(n-1) + 2\beta\mathbf{x}(n)\mathbf{x}^T(n) - \beta\mathbf{x}(n-L)\mathbf{x}^T(n) + \mathbf{x}(n)\mathbf{x}^T(n) - \beta\mathbf{x}(n)\mathbf{x}^T(n-L) \\ &= \mathbf{T}(n-1) + (2\beta\mathbf{x}(n) - \beta\mathbf{x}(n-L))\mathbf{x}^T(n) + \mathbf{x}(n)(\mathbf{x}(n) - \beta\mathbf{x}(n-L))^T\end{aligned} \tag{3.2}$$

Realize that equation (3.2) basically tells us that the matrix $\mathbf{T}(n)$ can be obtained recursively using a rank-2 update. In comparison (see Chapter 1), the RLS algorithm utilizes a rank-1 update for updating the covariance matrix. At this point, we invoke the matrix inversion lemma[2] (Sherman-Morrison-Woodbury identity) [7,8] given by

$$(\mathbf{A} + \mathbf{BCD}^T)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}(\mathbf{C}^{-1} + \mathbf{D}^T\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{D}^T\mathbf{A}^{-1} \tag{3.3}$$

Substituting $\mathbf{A} = \mathbf{T}(n-1)$, $\mathbf{B} = [(2\beta\mathbf{x}(n) - \beta\mathbf{x}(n-L)) \quad \mathbf{x}(n)]$, $\mathbf{C} = \mathbf{I}_{2x2}$, a 2x2 identity matrix and $\mathbf{D} = [\mathbf{x}(n) \quad (\mathbf{x}(n) - \beta\mathbf{x}(n-L))]$, we get the equation (3.2) in the same form as the LHS of equation (3.3). Therefore, the recursion for the inverse of $\mathbf{T}(n)$ becomes

[2] Notice that the matrix inversion lemma simplifies the computation of the matrix inverse only when the original matrix can be written using reduced rank updates.

$$\mathbf{T}^{-1}(n) = \mathbf{T}^{-1}(n-1) - \mathbf{T}^{-1}(n-1)\mathbf{B}(\mathbf{I}_{2x2} + \mathbf{D}^T\mathbf{T}^{-1}(n-1)\mathbf{B})^{-1}\mathbf{D}^T\mathbf{T}^{-1}(n-1) \quad (3.4)$$

Note that the computation of the above inverse is different than the conventional RLS algorithm. It requires the inversion of a 2x2 matrix $(\mathbf{I}_{2x2} + \mathbf{D}^T\mathbf{T}^{-1}(n-1)\mathbf{B})$ owing to the rank-2 update of $\mathbf{T}(n)$.The recursive estimator for $\mathbf{V}(n)$ is a simple correlation estimator given by

$$\mathbf{V}(n) = \mathbf{V}(n-1) + [(1+2\beta)d(n)\mathbf{x}(n) - \beta d(n)\mathbf{x}(n-L) - \beta d(n-L)\mathbf{x}(n)] \quad (3.5)$$

Using $\mathbf{T}^{-1}(n)$ and $\mathbf{V}(n)$, an estimate of the filter weight vector at iteration index n is

$$\mathbf{w}(n) = \mathbf{T}^{-1}(n)\mathbf{V}(n) \quad (3.6)$$

We will define a gain matrix analogous to the gain vector in the RLS case [14] as

$$\boldsymbol{\kappa}(n) = \mathbf{T}^{-1}(n-1)\mathbf{B}\left(\mathbf{I}_{2x2} + \mathbf{D}^T\mathbf{T}^{-1}(n-1)\mathbf{B}\right)^{-1} \quad (3.7)$$

Using the above definition, the recursive estimate for the inverse of $\mathbf{T}(n)$ becomes

$$\mathbf{T}^{-1}(n) = \mathbf{T}^{-1}(n-1) - \boldsymbol{\kappa}(n)\mathbf{D}^T\mathbf{T}^{-1}(n-1) \quad (3.8)$$

Once again, the above equation is analogous to the Ricatti equation for the RLS algorithm. Multiplying (3.7) from the right by $(\mathbf{I}_{2x2} + \mathbf{D}^T\mathbf{T}^{-1}(n-1)\mathbf{B})$, we obtain

$$\begin{aligned} &\boldsymbol{\kappa}(n)\left(\mathbf{I}_{2x2} + \mathbf{D}^T\mathbf{T}^{-1}(n-1)\mathbf{B}\right) = \mathbf{T}^{-1}(n-1)\mathbf{B} \\ &\boldsymbol{\kappa}(n) = \mathbf{T}^{-1}(n-1)\mathbf{B} - \boldsymbol{\kappa}(n)\mathbf{D}^T\mathbf{T}^{-1}(n-1)\mathbf{B} \\ &\boldsymbol{\kappa}(n) = \mathbf{T}^{-1}(n)\mathbf{B} \end{aligned} \quad (3.9)$$

In order to derive an update equation for the filter weights, we substitute the recursive estimate for $\mathbf{V}(n)$ in (3.6).

$$\mathbf{w}(n) = \mathbf{T}^{-1}(n)\mathbf{V}(n-1) + \mathbf{T}^{-1}(n)[(1+2\beta)d(n)\mathbf{x}(n) - \beta d(n)\mathbf{x}(n-L) - \beta d(n-L)\mathbf{x}(n)] \quad (3.10)$$

Using (3.8) and recognizing the fact that $\mathbf{w}(n-1) = \mathbf{T}^{-1}(n-1)\mathbf{V}(n-1)$ the above

equation can be reduced to

$$\begin{aligned}\mathbf{w}(n) &= \mathbf{w}(n-1) - \boldsymbol{\kappa}(n)\mathbf{D}^T\mathbf{w}(n-1) \\ &\quad + \mathbf{T}^{-1}(n)[(1+2\beta)d(n)\mathbf{x}(n) - \beta d(n)\mathbf{x}(n-L) - \beta d(n-L)\mathbf{x}(n)]\end{aligned} \tag{3.11}$$

Using the definition for $\mathbf{B} = [(2\beta\mathbf{x}(n) - \beta\mathbf{x}(n-L)) \quad \mathbf{x}(n)]$, we can easily see that

$$(1+2\beta)d(n)\mathbf{x}(n) - \beta d(n)\mathbf{x}(n-L) - \beta d(n-L)\mathbf{x}(n) = \mathbf{B}\begin{bmatrix} d(n) \\ d(n) - \beta d(n-L) \end{bmatrix} \tag{3.12}$$

From (3.9) and (3.12), the weight update equation simplifies to

$$\mathbf{w}(n) = \mathbf{w}(n-1) - \boldsymbol{\kappa}(n)\mathbf{D}^T\mathbf{w}(n-1) + \boldsymbol{\kappa}(n)\begin{bmatrix} d(n) \\ d(n) - \beta d(n-L) \end{bmatrix} \tag{3.13}$$

Note that the product $\mathbf{D}^T\mathbf{w}(n-1)$ is nothing but the matrix of the outputs $\begin{bmatrix} y(n) & y(n) - \beta y(n-L) \end{bmatrix}^T$, where $y(n) = \mathbf{x}^T(n)\mathbf{w}(n-1)$, $y(n-L) = \mathbf{x}^T(n-L)\mathbf{w}(n-1)$.

The apriori error matrix is defined as

$$\mathbf{e}(n) = \begin{bmatrix} d(n) - y(n) \\ d(n) - y(n) - \beta(d(n-L) - y(n-L)) \end{bmatrix} = \begin{bmatrix} e(n) \\ e(n) - \beta e(n-L) \end{bmatrix} \tag{3.14}$$

Using all the above definitions, we will formally state the weight update equation for the REW algorithm as

$$\mathbf{w}(n) = \mathbf{w}(n-1) + \boldsymbol{\kappa}(n)\mathbf{e}(n) \tag{3.15}$$

The overall complexity of (3.15) is $O(\mathrm{N}^2)$ which is comparable to the complexity of the RLS algorithm (this was achieved by using the matrix inversion lemma). Unlike the stochastic gradient algorithms that are easily affected by the eigenspread of the input data and the type of the stationary point solution (minimum, maximum or saddle), the REW algorithm is immune to these problems. This is because it inherently makes use of more information about the performance surface by computing the inverse of the Hessian

matrix $\mathbf{R} + \beta\mathbf{S}$. A summary of the REW algorithm is given below in Table 3-1.

Table 3-1. Outline of the REW Algorithm.

Initialize $\mathbf{T}^{-1}(0) = c\mathbf{I}$, c is a large positive constant

$\mathbf{w}(0) = \mathbf{0}$

At every iteration, compute

$$\mathbf{B} = [(2\beta\mathbf{x}(n) - \beta\mathbf{x}(n-L)) \quad \mathbf{x}(n)] \text{ and } \mathbf{D} = [\mathbf{x}(n) \quad (\mathbf{x}(n) - \beta\mathbf{x}(n-L))]$$

$$\boldsymbol{\kappa}(n) = \mathbf{T}^{-1}(n-1)\mathbf{B}\left(\mathbf{I}_{2x2} + \mathbf{D}^T\mathbf{T}^{-1}(n-1)\mathbf{B}\right)^{-1}$$

$$y(n) = \mathbf{x}^T(n)\mathbf{w}(n-1) \text{ and } y(n-L) = \mathbf{x}^T(n-L)\mathbf{w}(n-1)$$

$$\mathbf{e}(n) = \begin{bmatrix} d(n) - y(n) \\ d(n) - y(n) - \beta(d(n-L) - y(n-L)) \end{bmatrix} = \begin{bmatrix} e(n) \\ e(n) - \beta e(n-L) \end{bmatrix}$$

$$\mathbf{w}(n) = \mathbf{w}(n-1) + \boldsymbol{\kappa}(n)\mathbf{e}(n)$$

$$\mathbf{T}^{-1}(n) = \mathbf{T}^{-1}(n-1) - \boldsymbol{\kappa}(n)\mathbf{D}^T\mathbf{T}^{-1}(n-1)$$

The above derivation assumes stationary signals. For non-stationary signals, a forgetting factor is required for tracking. Inclusion of this factor in the derivation is trivial and is left out in this chapter. Also, note that the REW algorithm can be applied for any value of β. When β = -0.5, we know that AEC reduces to EWC and hence REW algorithm can be used for estimating the parameters in the presence of input white noise.

Extension of the REW Algorithm for Multiple Lags

In Chapter 2, we briefly mentioned the fact the AEC can be extended by including multiple lags in the cost function. It is easy to see that the extended AEC is given by

$$J(\mathbf{w}) = E[e^2(n)] + \sum_{L=1}^{L_{\max}} \beta E[e(n) - e(n-L)]^2 \tag{3.16}$$

where, $L_{\max}$ denotes the maximum number of lags utilized in the AEC cost function. It is not mandatory to use the same constant β for all the error lag terms. However, for the sake of simplicity, we assume single β value. The gradient of (3.16) with respect to the

weight vector **w**, is

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = -e(n)\mathbf{x}(n) - \beta \sum_{L=1}^{L_{\max}} [e(n) - e(n-L)][\mathbf{x}(n) - \mathbf{x}(n-L)] \tag{3.17}$$

Recall the following matrix definitions (restated here for clarity),

$$\begin{aligned}
\mathbf{R} &= E[\mathbf{x}(n)\mathbf{x}^T(n)] \\
\mathbf{S}_L &= E[(\mathbf{x}(n) - \mathbf{x}(n-L))(\mathbf{x}(n) - \mathbf{x}(n-L))^T] = 2(\mathbf{R} - \mathbf{R}_L) \\
\mathbf{R}_L &= E[\mathbf{x}(n)\mathbf{x}(n-L)^T + \mathbf{x}(n-L)\mathbf{x}(n)^T] \\
\mathbf{P} &= E[\mathbf{x}(n)d(n)] \\
\mathbf{P}_L &= E[\mathbf{x}(n)d(n-L) + \mathbf{x}(n-L)d(n)] \\
\mathbf{Q}_L &= E[(\mathbf{x}(n) - \mathbf{x}(n-L))(d(n) - d(n-L))^T] = 2\mathbf{P} - \mathbf{P}_L
\end{aligned} \tag{3.18}$$

Using the above definitions in (3.17) and equating the gradient to zero, we get the optimal extended AEC solution as shown below.

$$\mathbf{w}_* = (\mathbf{R} + \beta \sum_{L=1}^{L_{\max}} \mathbf{S}_L)^{-1} (\mathbf{P} + \beta \sum_{L=1}^{L_{\max}} \mathbf{Q}_L) \tag{3.19}$$

At first glance, the computational complexity of (3.19) seems to be $O(\mathrm{N}^3)$. But, the symmetric structure of the matrices involved can be exploited to lower the complexity. Once again, we resort to the matrix inversion lemma as before and deduce a lower $O(\mathrm{N}^2)$ complexity algorithm. Realize that the optimal extended AEC solution at any time instant *n* will be

$$\mathbf{w}(n) = \mathbf{T}^{-1}(n)\mathbf{V}(n) \tag{3.20}$$

where, $\mathbf{T}(n) = \mathbf{R}(n) + \beta \sum_{L=1}^{L_{\max}} \mathbf{S}_L(n)$ and $\mathbf{V}(n) = \mathbf{P}(n) + \beta \sum_{L=1}^{L_{\max}} \mathbf{Q}_L(n)$ as before. The estimator for the vector $\mathbf{V}(n)$ will be a simple recursive correlator.

$$\mathbf{V}(n) = \mathbf{V}(n-1) + [(1 + 2\beta L_{\max})d(n)\mathbf{x}(n) - \beta \sum_{L=1}^{L_{\max}} d(n)\mathbf{x}(n-L) - \beta d(n-L)\mathbf{x}(n)] \tag{3.21}$$

The matrix $\mathbf{T}(n)$ can be estimated recursively as follows.

$$\begin{aligned}\mathbf{T}(n) &= \mathbf{T}(n-1) + (1+2\beta L_{\max})\mathbf{x}(n)\mathbf{x}^T(n) - \beta\sum_{L=1}^{L_{\max}}\mathbf{x}(n-L)\mathbf{x}^T(n) - \beta\sum_{L=1}^{L_{\max}}\mathbf{x}(n)\mathbf{x}^T(n-L) \\ &= \mathbf{T}(n-1) + 2\beta L_{\max}\mathbf{x}(n)\mathbf{x}^T(n) - \beta\sum_{L=1}^{L_{\max}}\mathbf{x}(n-L)\mathbf{x}^T(n) + \\ &\quad \mathbf{x}(n)\mathbf{x}^T(n) - \beta\sum_{L=1}^{L_{\max}}\mathbf{x}(n)\mathbf{x}^T(n-L) \\ \mathbf{T}(n) &= \mathbf{T}(n-1) + \left(2\beta L_{\max}\mathbf{x}(n) - \beta\sum_{L=1}^{L_{\max}}\mathbf{x}(n-L)\right)\mathbf{x}^T(n) + \mathbf{x}(n)\left(\mathbf{x}(n) - \beta\sum_{L=1}^{L_{\max}}\mathbf{x}(n-L)\right)^T\end{aligned} \tag{3.21}$$

Now, the matrices **A**, **B**, **C** and **D** used in the inversion lemma in equation (3.3) are defined as follows.

$$\begin{aligned}\mathbf{A} &= \mathbf{T}(n-1) \\ \mathbf{B} &= \left[\left(2\beta L_{\max}\mathbf{x}(n) - \beta\sum_{L=1}^{L_{\max}}\mathbf{x}(n-L)\right) \quad \mathbf{x}(n)\right] \\ \mathbf{C} &= \mathbf{I}_{2\times 2} \\ \mathbf{D} &= \left[\mathbf{x}(n) \quad \left(\mathbf{x}(n) - \beta\sum_{L=1}^{L_{\max}}\mathbf{x}(n-L)\right)\right]\end{aligned} \tag{3.22}$$

The only differences from the previous definitions lie in the expressions for the **B** and **D** matrices that now require an inner loop running up to $L_{\max}$. The rest of the procedure remains the same as before. Once again, by the proper application of the matrix inversion lemma, we were able to reduce the complexity of the matrix inversion to $O(\mathrm{N}^2)$ by recursively computing the inverse in a way that we only require an inversion of a simple 2x2 matrix. This measure of complexity does not include the computations involved in building the **B** and **D** matrices. However, typically, the maximum number of lags will be smaller than the length of the adaptive filter. Therefore, the additional overhead incurred in the estimation of **B** and **D** matrices will not result in a significant change in the overall complexity.

Relationship to the Recursive Instrumental Variables Method

The previously derived REW algorithm for the single lag case has a structure similar to the Instrumental Variables (IV) method. The IV method has its origins in statistics and was apparently proposed by Reiersøl [122]. Over a period of time, it has been adapted to model dynamical systems in control engineering. Lot of work in the applications of IV to control engineering problems has been done by Wong, Polak [123] and Young [124-126]. Recent advances in IV methods for system identification and control have been mainly due to Söderström and Stoica [32,93]. It is beyond the scope of this dissertation to summarize the applications and impacts of IV in various engineering problems. For more details, refer to [32].

Basically, IV can be viewed as an extension to the standard Least Squares regression and can be used to estimate the parameters in white noise once the model order is known. The fundamental principle is to choose delayed regression vectors known as *instruments* that are uncorrelated with the additive white noise. IV can also be extended to handle colored noise situations. This will be exclusively handled in Chapter 5. For now, the discussion will be strictly limited to the white noise scenario. Mathematically speaking, the IV method computes the solution

$$\mathbf{w}_{IV} = E[\mathbf{x}(n)\mathbf{x}(n-\Delta)^T]^{-1} E[\mathbf{x}(n)d(n-\Delta)] \tag{3.23}$$

where, the lag Δ is chosen such that the outer product of the regression vector $\mathbf{x}(n)$ with the lagged regression vector $\mathbf{x}(n\text{-}\Delta)$ result in a matrix that is independent of the additive white noise components $v(n)$. In comparison, the REW solution is given by $\mathbf{w}_* = \mathbf{R}_L^{-1}\mathbf{P}_L$. Notice that in REW solution, the matrix $\mathbf{R}_L$ is symmetric and Toeplitz [8], which is very much desirable and we exploit this fact to derive an elegant minor components based

algorithm in the next section. Thus, in effect the IV method can be considered as a special case of the REW algorithm obtained by removing the symmetric terms in $\mathbf{R}_L$ and $\mathbf{P}_L$. We will compare the performances of REW and IV methods later in this chapter.

Recursive EWC Algorithm Based on Minor Components Analysis

Till now, we focused on a Newton type algorithm to compute the optimal AEC solution. Although the algorithm is fast converging, the convergence of the algorithm can be sensitive to the ill-conditioning of the Hessian matrix $\mathbf{R}(n) + \beta\mathbf{S}(n)$ which can happen at the first few iterations. Alternatively, we can explore the idea of using the minor components analysis (MCA) to derive a recursive algorithm similar to the TLS algorithm for MSE. We call this algorithm as EWC-TLS algorithm. As the name suggests, this algorithm can be used only for the case with β = -0.5, which defaults the augmented error criterion to error whitening criterion. Recall that the TLS problem in general, solves an over-determined set of linear equations of the form $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A} \in \Re^{m \times n}$ is the data matrix, $\mathbf{b} \in \Re^{m}$ is the desired vector, and $\mathbf{x} \in \Re^{n}$ is the parameter vector and m denotes the number of different observation vectors each of dimension n[41]. Alternatively, the linear equations can be written in the form $[\mathbf{A};\mathbf{b}][\mathbf{x}^T;-1] = \mathbf{0}$, where $[\mathbf{A};\mathbf{b}]$ denotes an augmented data matrix. When $[\mathbf{A};\mathbf{b}]$ is a symmetric square matrix, it can be shown that the TLS solution is simply given by

$$[\mathbf{x};-1] = -\mathbf{v}_{n+1} / \mathbf{v}_{n+1,n+1} \tag{3.24}$$

where, $\mathbf{v}_{n+1,n+1}$ is the last element of the minor eigenvector $\mathbf{v}_{n+1}$. In the case of EWC, it is easy to show that the augmented data matrix [127,128] (analogous to $[\mathbf{A};\mathbf{b}]$) is

$$\mathbf{G} = \begin{bmatrix} \mathbf{R}_L & \mathbf{P}_L \\ \mathbf{P}_L^T & 2\rho_d(L) \end{bmatrix} \tag{3.25}$$

The term $\rho_d(L)$ in (3.25) denotes the autocorrelation of the desired signal at lag *L*. It is important to note that the matrix (3.25) is square symmetric due to the symmetry of $\mathbf{R}_L$. Hence, the eigenvectors of **G** are all real which is highly desirable. Also, it is important to note the fact that (3.25) still holds even with noisy data as the entries of **G** are unaffected by the noise terms. In the infinite-sample case, the matrix **G** is not full rank and we can immediately see that one of the eigenvalues of (3.25) is zero. In the finite-sample case, the goal would be to find the eigenvector corresponding to the minimum absolute eigenvalue (finite samples also imply that $\mathbf{G}^{-1}$ exists). Since the eigenvalues of **G** can be both positive and negative, typical iterative gradient or even some fixed-point type algorithms tend to become unstable. A workaround would be to use the matrix $\mathbf{G}^2$ instead of **G**. This will obviate the problem of having mixed eigenvalues while still preserving the eigenvectors. However, the squaring operation is good only if the eigenvalues of **G** are well separated. Otherwise, the smaller eigenvalues blend together making the estimation of the minor component of $\mathbf{G}^2$ more difficult. Also, the squaring operation creates additional overhead, thereby negating any computational benefits offered by the fixed point type PCA solutions as discussed in Appendix A.

So, we propose to use the inverse iteration method for estimating the minor eigenvector of **G** [8]. If $\mathbf{w}(n) \in \Re^{N+1}$ denotes the estimate of the minor eigenvector corresponding to the smallest absolute eigenvalue at time instant *n*, then the estimate at the $(n+1)^{\text{th}}$ instant is given by

$$\begin{aligned} \mathbf{w}(n+1) &= \mathbf{G}(n+1)^{-1}\mathbf{w}(n) \\ \mathbf{w}(n+1) &= \frac{\mathbf{w}(n+1)}{\|\mathbf{w}(n+1)\|} \end{aligned} \tag{3.26}$$

The term $\mathbf{G}(n+1)$ denotes the estimate of the augmented data matrix **G** (equation (3.25))

at the $(n+1)^{th}$ instant. It is easy to see that $\mathbf{G}(n)$ can be recursively estimated as $\mathbf{G}(n) = \mathbf{G}(n-1) + \boldsymbol{\psi}(n)\boldsymbol{\psi}(n-L)^T + \boldsymbol{\psi}(n-L)\boldsymbol{\psi}(n)^T$ where, $\boldsymbol{\psi}(n) = [\mathbf{x}(n); d(n)]$ is the concatenated vector of the input and desired response. Now, we can invoke the inversion lemma as before and obtain a recursive $O(n^2)$ estimate for matrix inversion in (3.26). The details of this derivation are trivial and omitted here. Once the minor component estimate converges i.e., $\mathbf{w}(n) \to \mathbf{v}$, the EWC-TLS solution is simply given by equation (3.24). Thus, the overall complexity of the EWC-TLS algorithm is still $O(n^2)$ which is the same as the REW algorithm. However, we have observed through simulations that, the EWC-TLS method converges faster than the EWC-REW while preserving the accuracy of the parameter estimates.

Experimental Results

We will now show the simulation results with the Newton type algorithms for AEC. Specifically, our objective is to show the superior performance of the proposed criterion and the associated algorithms in the problem of system identification with noisy input data.

Estimation of System Parameters in White Noise Using REW

The REW algorithm can be used effectively to solve the system identification problem in noisy environments. As we have seen before, setting the value of $\beta = -0.5$, noise immunity can be gained for parameter estimation. We generated a purely white Gaussian random noise of length 50,000 samples and added this to a colored input signal. The white noise signal is uncorrelated with the input signal. The noise free, colored, input signal was filtered by the unknown reference filter, and this formed the desired signal for the adaptive filter. Since, the noise in the desired signal would be averaged out for both

RLS and REW algorithms, we decided to use the clean desired signal itself. This will bring out only the effects of input noise on the filter estimates. Also, the noise added to the clean input is uncorrelated with the desired signal. In the experiment, we varied the Signal-to-Noise-Ratio (SNR) in the range –10dB to +10dB. The number of desired filter coefficients was also varied from 4 to 12. We then performed 100 Monte Carlo runs and computed the normalized error vector norm given by

$$error = 20\log 10\left[\frac{\|\mathbf{w}_T - \mathbf{w}_*\|}{\|\mathbf{w}_T\|}\right] \tag{3.27}$$

where, $\mathbf{w}_*$ is the weight vector estimated by the REW algorithm with $\beta = -0.5$ after 50,000 iterations or one complete presentation of the input data and $\mathbf{w}_T$ is the true weight vector. In order to show the effectiveness of the REW algorithm, we performed Monte Carlo runs using the RLS algorithm on the same data to estimate the filter coefficients. Further, we also evaluated the analytical TLS solution for each case Figure 3-1 shows a histogram plot of the normalized error vector norm given in (3.27) for all the three methods. It is clear that the REW algorithm was able to perform better than the RLS at various SNR and tap length settings. In the high SNR cases, there is not much of a difference between RLS and REW results. However, under noisy circumstances, the reduction in the parameter estimation error with REW is orders of magnitude more when compared with RLS. Also, the RLS algorithm results in a rather useless zero weight vector, i.e., $\mathbf{w} = \mathbf{0}$ when the SNR is lower than –10dB. On the other hand, TLS performs well only in the cases when the noise variances in the input and desired signals are the same. This is in conformance with the well-known theoretical limitations of the TLS algorithm.

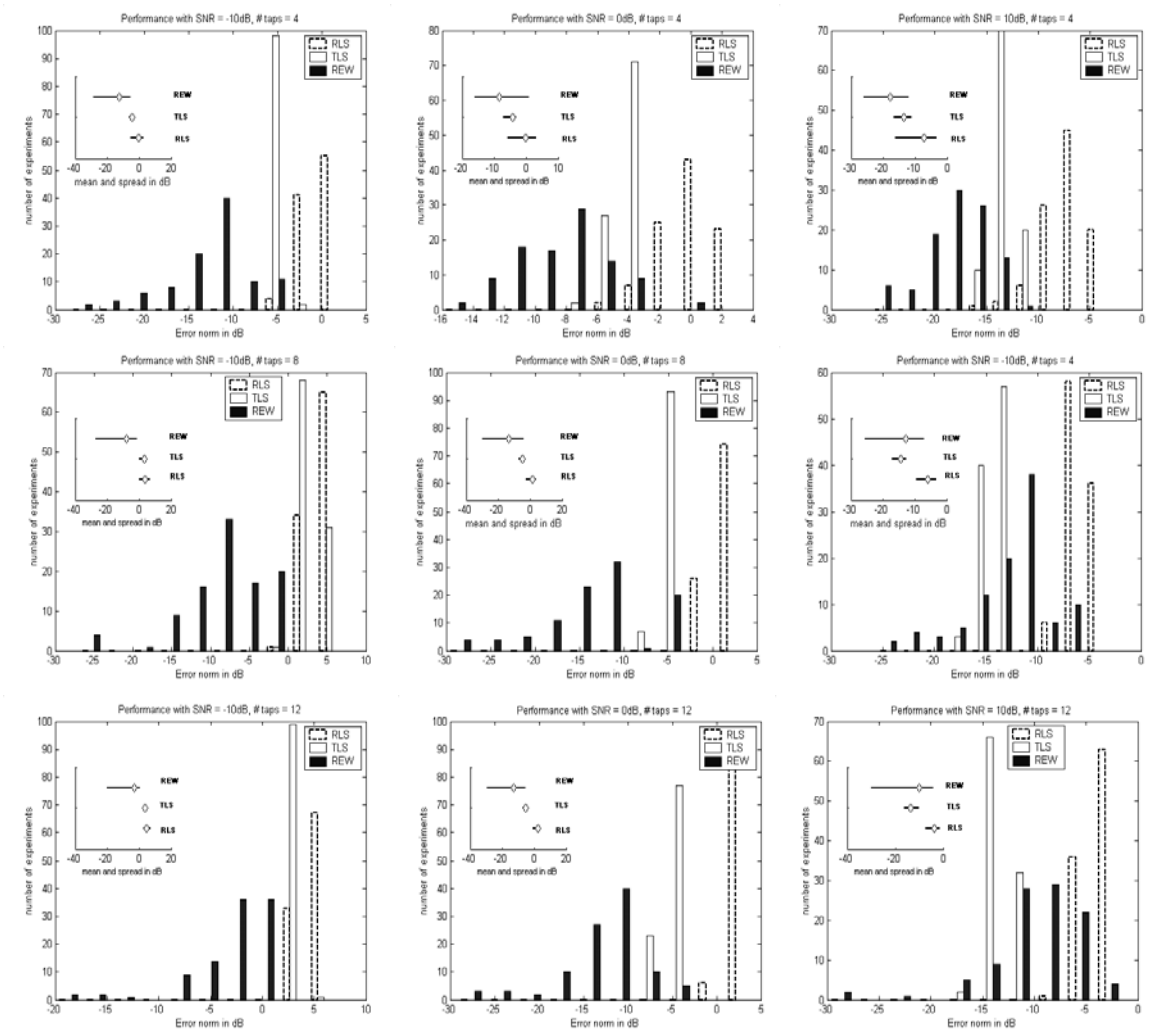
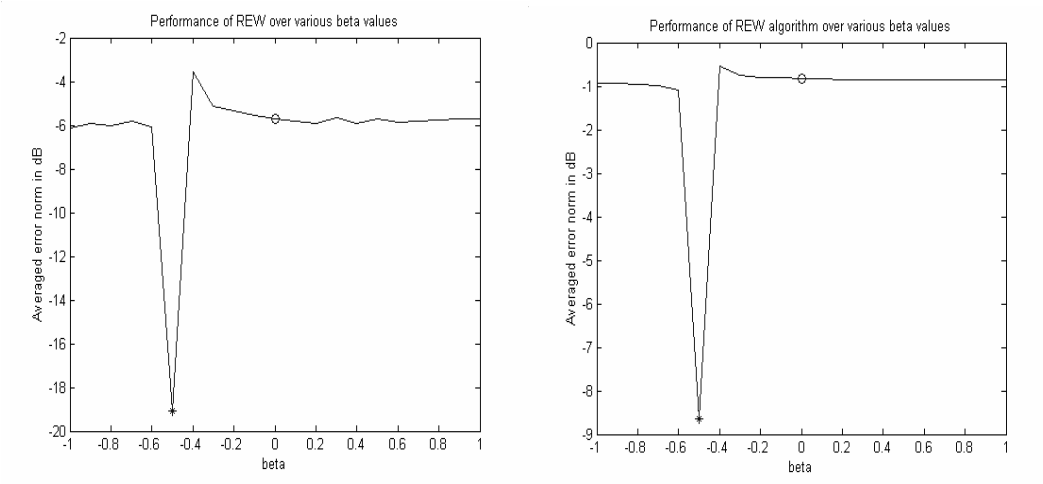


Figure 3-1. Histogram plots showing the error vector norm for EWC-LMS, LMS algorithms and the numerical TLS solution.

Effect of β and Weight Tracks of REW Algorithm

Since we have a free parameter β to choose, it would be worthwhile to explore the effect of β on the AEC parameter estimates. The SNR of the input signal was fixed at values 0dB and –10dB, the number of filter taps was set to 4 and the desired signal was noise free as before. We performed 100 Monte Carlo experiments and analyzed the average error vector norm values for $-1 \le \beta \le 1$. The results of the experiment are shown in Figure 3-2. Notice that there is a dip at $\beta = -0.5$ (indicated by a "*" in the figure) and



(a) (b)

Figure 3-2. Performance of REW algorithm (a) SNR = 0dB and (b) SNR = -10 over various beta values.

this clearly gives us the minimum estimation error. This corresponds to the EWC solution. For $\beta = 0$, (indicated by a "o" in the figure) the REW algorithm reduces to the regular RLS giving a fairly significant estimation error.

Next the parameter β is set to –0.5 and SNR to 0dB, and the weight tracks are estimated for the REW and the RLS algorithms. Figure 3-3 shows the averaged weight tracks for both REW and RLS algorithms averaged over 50 Monte Carlo trials. Asterisks on the plots indicate the true parameters. The tracks for the RLS algorithm are smoother, but they converge to wrong values, which we have observed quite consistently. The weight tracks for the REW algorithm are noisier compared to those of the RLS, but they eventually converge to values very close to the true weights. This brings us to an important issue of estimators viz., bias and the variance. The RLS algorithm has a reduced variance because of the positive-definiteness of the covariance matrix **R**(*n*). However, the RLS solution remains asymptotically, biased in the presence of noisy input.

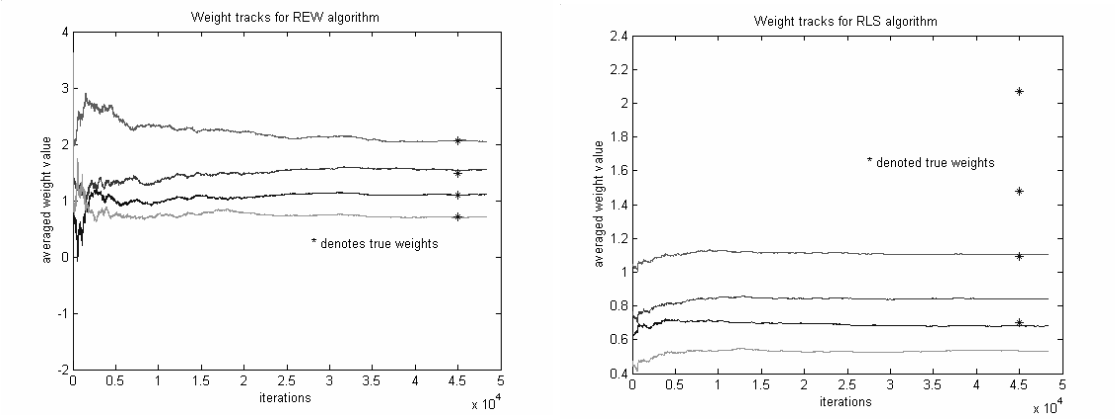


Figure 3-3. Weight tracks for REW and RLS algorithms.

On the other hand, REW algorithm produces zero bias, but the variance can be high owing to the conditioning of the Hessian matrix. However, this variance diminishes with increasing number of samples.

The noisy initial weight tracks of the REW algorithm may be attributed to the ill conditioning that is mainly caused by the smallest eigenvalue of the estimated Hessian matrix, which is $\mathbf{R}(n)+\beta\mathbf{S}(n)$. The same holds true for the RLS algorithm, where the minimum eigenvalue of $\mathbf{R}(n)$ affects the sensitivity [14]. The instability issues of the RLS algorithm during the initial stages of adaptation have been well studied in literature and effects of round off error have been analyzed and many solutions have been proposed to make the RLS algorithm robust to such effects [129]. Similar analysis on the REW algorithm is yet to be done and this would be addressed in future work on the topic.

Performance Comparisons between REW, EWC-TLS and IV methods

In this example, we will contrast the performances of the REW, EWC-TLS and the Instrumental Variables (IV) method in a 4-tap system identification problem with noisy data. The input signal is colored and corrupted with white noise (input SNR was set at 5dB) and the desired signal SNR is 10dB. For the IV method, we chose the delayed input

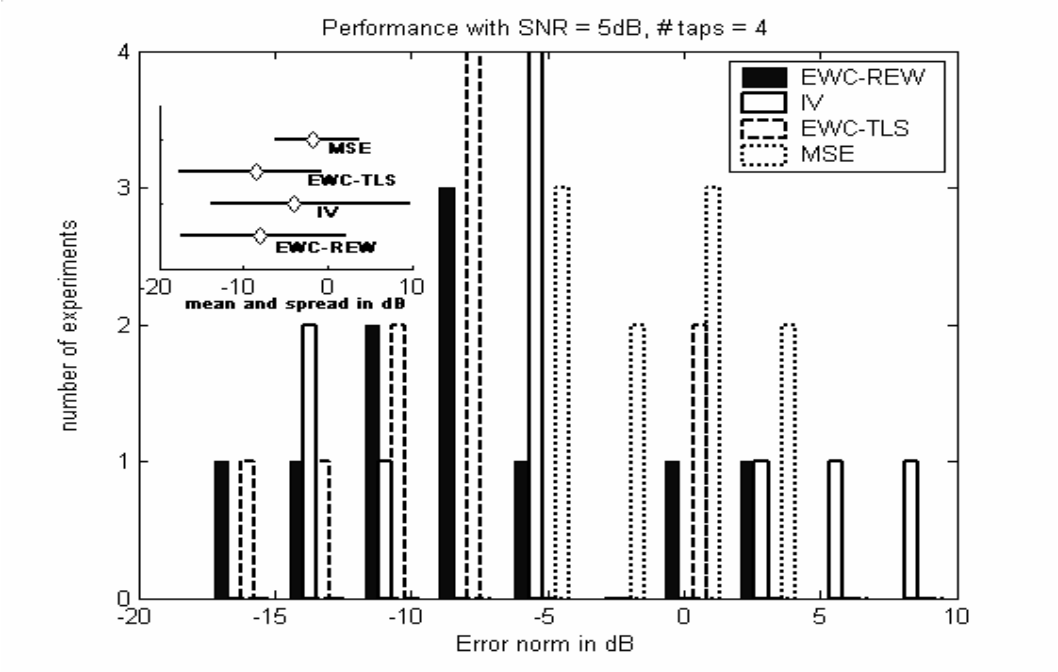


Figure 3-4. Histogram plots showing the error vector norms for all the methods.

vector $\mathbf{x}(n-\Delta)$ as the instrument and the lag Δ was chosen to be four, which is the length of the true filter. Once again, the performance metric was chosen as the error vector norm in dB given by equation (3.27). Figure 3-4 shows the error histograms for REW, EWC-TLS, IV and the optimal Wiener solutions. EWC-TLS and REW algorithms outperform the Wiener MSE solution. The IV method also produces better results than the Wiener solution. Amongst the EWC solutions, we obtained better results with the EWC-TLS algorithm (equations 3.24 and 3.26) than REW. However, both EWC-TLS and REW outperformed IV method. This may be partially attributed to the conditioning of the matrices involved in the estimation of the REW and IV methods. Further theoretical analysis is required to quantify the effects of conditioning and symmetric Toeplitz structure of $\mathbf{R}_L$. In Figure 3-5, we show the angle between the estimated minor eigenvector and the true eigenvector of the augmented data matrix **G** for a random single

trial in scenarios with and without noise. Notice that the rates of convergence are very much different. It is well known that the rate of convergence for inverse iteration method is given by the ratio $|\lambda_1/\lambda_2|$, where $|\lambda_1|^{-1}$ is the largest eigenvalue of $\mathbf{G}^{-1}$ and $|\lambda_2|^{-1}$ is the second largest eigenvalue of $\mathbf{G}^{-1}$ [8]. Faster convergence can be seen in the noiseless case owing to the huge $|\lambda_1/\lambda_2|$ ratio.

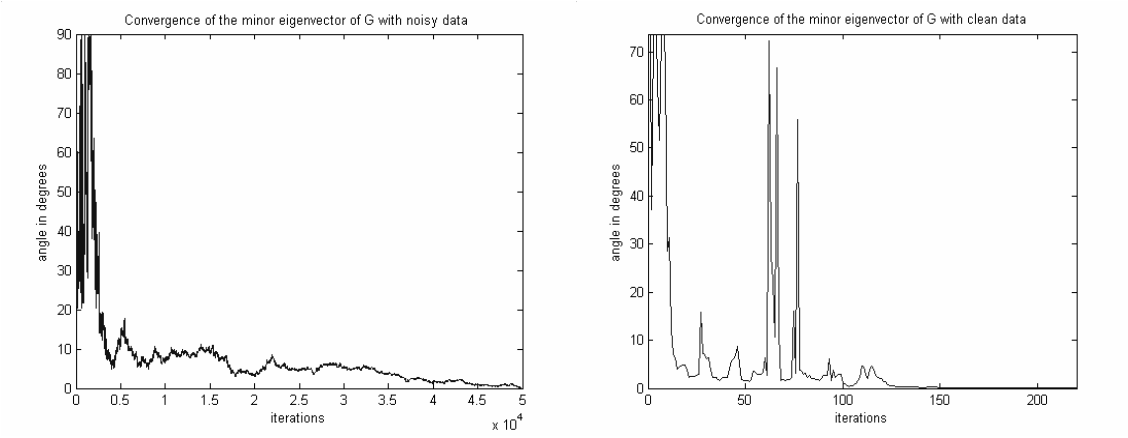


Figure 3-5. Convergence of the minor eigenvector of **G** with (a) noisy data and (b) clean data.

Summary

In this chapter, we derived recursive Newton type algorithm to estimate the optimal AEC solution. First, the Recursive Error Whitening (REW) algorithm was derived using the analytical AEC solution and the matrix inversion lemma. The well-known RLS algorithm for MSE becomes a special case of the REW algorithm. Further, a Total Least-Squares based EWC algorithm called EWC-TLS was proposed. This algorithm works with β = -0.5 and can be easily applied to estimate parameters in the presence of white noise. A fixed-point minor components extraction algorithm was developed using the inverse iteration method. Other fixed-point or gradient-based methods cannot be used because of the indefiniteness (matrix with mixed eigenvalues make the algorithms locally

unstable) of the matrix involved in the EWC-TLS formulation. The computational complexity of the above mentioned algorithms is $O(N^2)$. We briefly explored an extension of the Newton type algorithm for the extended AEC with multiple lags. Effective usage of the matrix inversion lemma can cut the complexity of the extended REW algorithm to $O(N^2)$.

In the later half of the chapter, we discussed the performance of the algorithms in the problem of system identification in the presence of additive white noise. The proposed recursive algorithms outperform the RLS and the analytical MSE TLS solutions. We also showed the simulation results with the EWC-TLS algorithm and quantitatively compared the performance with the well-known IV method.

Although the recursive EWC algorithms presented in this chapter are fast converging and sample efficient, the complexity of $O(N^2)$ can be high for many applications involving low power designs. Additionally, the recursive algorithms can exhibit limited performance in non-stationary conditions if the forgetting factors are not chosen properly. This motivates us to explore stochastic gradient (and its variants) algorithms for estimating the optimal AEC solution. Chapter 5 will describe these algorithms and also highlight other benefits of the stochastic algorithms over their Newton type counterparts.

CHAPTER 4
STOCHASTIC GRADIENT ALGORITHMS FOR AEC

Introduction

Stochastic gradient algorithms have been at the forefront in optimizing quadratic cost functions like the MSE. Owing to the presence of a global minimum in quadratic performance surfaces, gradient algorithms can elegantly accomplish the task of reaching the optimal solution at minimal computational cost. In this chapter, we will derive the stochastic gradient algorithms for the AEC. Since the AEC performance surface is a weighted sum of quadratics, we can expect that difficulties will arise. However, we will show that using some simple optimization tricks, we can overcome these difficulties in an elegant manner.

Derivation of the Stochastic Gradient AEC-LMS Algorithm

Assume that we have a noisy training data set of the form $(\mathbf{x}(n), d(n))$, where $\mathbf{x}(n) \in \Re^m$ is the input and $d(n) \in \Re$ is the output of a linear system with coefficient vector $\mathbf{w}_T$. The goal is to estimate the parameter vector $\mathbf{w}_T$ using the augmented error criterion. We know that the AEC cost function is given by

$$J(\mathbf{w}) = E(e^2(n)) + \beta E(\dot{e}^2(n)) \tag{4.1}$$

where, $\dot{e}(n) = e(n) - e(n-L)$, $\mathbf{w}$ is the estimate of the parameter vector and $L \geq m$, the size of the input vector. For convenience, we will restate the following definitions, $\dot{\mathbf{x}}(n) = \mathbf{x}(n) - \mathbf{x}(n-L)$, $\dot{d}(n) = d(n) - d(n-L)$, $\mathbf{R} = E[\mathbf{x}(n)\mathbf{x}^T(n)]$, $\mathbf{S} = E[\dot{\mathbf{x}}(n)\dot{\mathbf{x}}^T(n)]$, $\mathbf{P} = E[\mathbf{x}(n)d(n)]$ and $\mathbf{Q} = E[\dot{\mathbf{x}}(n)\dot{d}(n)]$. Using these definitions, we can rewrite the cost

function in (4.1) as

$$J(\mathbf{w}) = E[d^2(n)] + \beta E[\dot{d}^2(n)] + \mathbf{w}^T(\mathbf{R} + \beta\mathbf{S})\mathbf{w} - 2(\mathbf{P} + \beta\mathbf{Q})^T\mathbf{w} \tag{4.2}$$

It is easy to see that both $E[e^2(n)]$ and $E[\dot{e}^2(n)]$ have parabolic performance surfaces as their Hessians have positive eigenvalues. However, the value of β can invert the performance surface of $E[\dot{e}^2(n)]$. For $\beta > 0$, the stationary point is always a global minimum and the gradient given by equation (4.2) can be written as the sum of the individual gradients as shown below.

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = 2(\mathbf{R} + \beta\mathbf{S})\mathbf{w} - 2(\mathbf{P} + \beta\mathbf{Q}) = 2(\mathbf{R}\mathbf{w} - \mathbf{P}) + 2\beta(\mathbf{S}\mathbf{w} - \mathbf{Q}) \tag{4.3}$$

The above gradient can be approximated by the stochastic instantaneous gradient by removing the expectation operators.

$$\frac{\partial J(\mathbf{w}(n))}{\partial \mathbf{w}(n)} \approx -\left[e(n)\mathbf{x}(n) + \beta\dot{e}(n)\dot{\mathbf{x}}(n)\right] \tag{4.4}$$

The goal is to minimize the cost function and hence using steepest descent, we can write the weight update for the stochastic AEC-LMS algorithm for $\beta > 0$ as

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)\left(e(n)\mathbf{x}(n) + \beta\dot{e}(n)\dot{\mathbf{x}}(n)\right) \tag{4.5}$$

where $\eta(n) > 0$ is a finite step-size parameter that controls convergence. For $\beta < 0$, the stationary point is still unique, but it can be a saddle point, global maximum or a global minimum depending on the value β. Evaluating the gradient as before and using the instantaneous gradient, we get the AEC-LMS algorithm for $\beta < 0$.

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)\left(e(n)\mathbf{x}(n) - |\beta|\dot{e}(n)\dot{\mathbf{x}}(n)\right) \tag{4.6}$$

where, $\eta(n)$ is again a small step-size. However, there is no guarantee that the above update rules will be stable for all choices of step-sizes. Although, equations (4.5) and

(4.6) are identical, we will use $-|\beta|$ in the update equation (4.6) to analyze the convergence of the algorithm specifically for $\beta < 0$. The reason for the separate analysis is that the convergence characteristics of (4.5) and (4.6) are very different.

Convergence Analysis of AEC-LMS Algorithm

Theorem 4.1. The stochastic AEC algorithms asymptotically converge in the mean to the optimal solution given by

$$\begin{aligned} \mathbf{w}_* &= (\mathbf{R} + \beta\mathbf{S})^{-1}(\mathbf{P} + \beta\mathbf{Q}), \quad \beta > 0 \\ \mathbf{w}_* &= (\mathbf{R} + \beta\mathbf{S})^{-1}(\mathbf{P} - |\beta|\mathbf{Q}), \quad \beta < 0 \end{aligned} \tag{4.7}$$

We will make the following mild assumptions typically applied to stochastic approximation algorithms [79-81,84] that can be easily satisfied.

1. The input vectors $\mathbf{x}(n)$ are derived from at least a wide sense stationary (WSS) colored random signal with positive definite autocorrelation matrix $\mathbf{R} = E[\mathbf{x}(n)\mathbf{x}^T(n)]$
2. The matrix $\mathbf{R}_L = E[\mathbf{x}(n)\mathbf{x}^T(n-L) + \mathbf{x}(n-L)\mathbf{x}^T(n)]$ exists and has full rank
3. The sequence of weight vectors $\mathbf{w}(n)$ is bounded with probability 1
4. The update functions $h(\mathbf{w}(n)) = e(n)\mathbf{x}(n) + \beta\dot{e}(n)\dot{\mathbf{x}}(n)$ for $\beta > 0$ and $h(\mathbf{w}(n)) = e(n)\mathbf{x}(n) - |\beta|\dot{e}(n)\dot{\mathbf{x}}(n)$ for $\beta < 0$ exist and are continuously differentiable with respect to $\mathbf{w}(n)$, and their derivatives are bounded in time.
5. Even if $h(\mathbf{w}(n))$ has some discontinuities a mean update vector $\bar{h}(\mathbf{w}(n)) = \lim_{n\to\infty} E[h(\mathbf{w}(n))]$ exists.

Assumption A.1 is easily satisfied. A.2 requires that the input signal have sufficient correlation with itself for at least L lags.

Proof of AEC-LMS Convergence for β > 0

We will first consider the update equation in (4.5) which is the stochastic AEC-LMS algorithm for $\beta > 0$. Without loss of generality, we will assume that the input

vectors $\mathbf{x}(n)$ and their corresponding desired responses $d(n)$ are noise-free. The mean update vector $\bar{h}(\mathbf{w}(n))$ is given by

$$\bar{h}(\mathbf{w}(n)) = \frac{d\mathbf{w}(t)}{dt} = E\left[e(n)\mathbf{x}(n) + \beta\dot{e}(n)\dot{\mathbf{x}}(n)\right] = \mathbf{R}\mathbf{w}(n) - \mathbf{P}(n) + \beta(\mathbf{S}\mathbf{w}(n) - \mathbf{Q}(n)) \tag{4.8}$$

The stationary point of the ordinary differential equation (ODE) in (4.8) is given by

$$\mathbf{w}_* = \left(\mathbf{R} + \beta\mathbf{S}\right)^{-1}\left(\mathbf{P} + \beta\mathbf{Q}\right) \tag{4.9}$$

We will define the error vector at time instant *n* as $\boldsymbol{\xi}(n) = \mathbf{w}_* - \mathbf{w}(n)$. Therefore

$$\boldsymbol{\xi}(n+1) = \boldsymbol{\xi}(n) - \eta(n)\left[e(n)\mathbf{x}(n) + \beta\dot{e}(n)\dot{\mathbf{x}}(n)\right] \tag{4.10}$$

and the norm of the error vector at time *n*+1 is simply

$$\begin{aligned}\left\|\boldsymbol{\xi}(n+1)\right\|^2 = \left\|\boldsymbol{\xi}(n)\right\|^2 - 2\eta(n)\left[\boldsymbol{\xi}^T(n)e(n)\mathbf{x}(n) + \beta\boldsymbol{\xi}^T(n)\dot{e}(n)\dot{\mathbf{x}}(n)\right] + \\ \eta^2(n)\left\|e(n)\mathbf{x}(n) + \beta\dot{e}(n)\dot{\mathbf{x}}(n)\right\|^2\end{aligned} \tag{4.11}$$

Imposing the condition that $\left\|\boldsymbol{\xi}(n+1)\right\|^2 < \left\|\boldsymbol{\xi}(n)\right\|^2$ for all *n*, we get an upper bound on the time varying step-size parameter $\eta(n)$ which is given by

$$\eta(n) < \frac{2\left[\boldsymbol{\xi}^T(n)e(n)\mathbf{x}(n) + \beta\boldsymbol{\xi}^T(n)\dot{e}(n)\dot{\mathbf{x}}(n)\right]}{\left\|e(n)\mathbf{x}(n) + \beta\dot{e}(n)\dot{\mathbf{x}}(n)\right\|^2} \tag{4.12}$$

Simplifying the above equation using the fact that $\boldsymbol{\xi}^T(n)\mathbf{x}(n) = e(n)$ and $\boldsymbol{\xi}^T(n)\dot{\mathbf{x}}(n) = \dot{e}(n)$, we get

$$\eta(n) < \frac{2\left[e^2(n) + \beta\dot{e}^2(n)\right]}{\left\|e(n)\mathbf{x}(n) + \beta\dot{e}(n)\dot{\mathbf{x}}(n)\right\|^2} \tag{4.13}$$

which is a more practical upper bound on the step-size as it can be directly estimated from the input and desired data. As an observation, notice that if $\beta = 0$, then, the bound in (4.13) reduces to

$$\eta(n) < \frac{2}{\|\mathbf{x}(n)\|^2} \tag{4.14}$$

which, when included in the update equation, reduces to a variant of the Normalized LMS (NLMS) algorithm [14]. In general, if the step-size parameter is chosen according to the bound given by (4.13), then the norm of the error vector $\boldsymbol{\xi}(n)$ is a monotonically decreasing sequence converging asymptotically to zero, i.e., $\lim_{n\to\infty}\|\boldsymbol{\xi}(n)\|^2 \to 0$ which implies that $\lim_{n\to\infty}\mathbf{w}(n) \to \mathbf{w}_*$ (see Appendix G for details). In addition, the upper bound on the step-size ensures that the weights are always bound with probability one satisfying the assumption A.3 we made before. Thus the weight vector $\mathbf{w}(n)$ converges asymptotically to $\mathbf{w}_*$, which is the only stable stationary point of the ODE in (4.8). Note that (4.5) is an $O(m)$ algorithm.

Proof of AEC-LMS Convergence for $\beta < 0$

We analyze the convergence of the stochastic gradient algorithm for $\beta < 0$ in the presence of white noise because this is the relevant case ($\beta = -0.5$ eliminates the bias due to noise added to the input). From (4.6), the mean update vector $\bar{h}(\mathbf{w}(n))$ is given by,

$$\bar{h}(\mathbf{w}(n)) = \frac{d\mathbf{w}(t)}{dt} = E\left[e(n)\mathbf{x}(n) - |\beta|\dot{e}(n)\dot{\mathbf{x}}(n)\right] = \mathbf{R}\mathbf{w}(n) - \mathbf{P}(n) - |\beta|(\mathbf{S}\mathbf{w}(n) - \mathbf{Q}(n)) \tag{4.15}$$

As before, the stationary point of this ODE is

$$\mathbf{w}_* = \left(\mathbf{R} - |\beta|\mathbf{S}\right)^{-1}\left(\mathbf{P} - |\beta|\mathbf{Q}\right) \tag{4.16}$$

The eigenvalues of $\mathbf{R} - |\beta|\mathbf{S}$ decide the nature of the stationary point. If they are all positive, then we have a global minimum and if they are all negative, we have a global maximum. In these two cases, the stochastic gradient algorithm in (4.6) with proper fixed

sign step-size would converge to the stationary point, which would be stable. However, we know that the eigenvalues of $\mathbf{R}-|\beta|\mathbf{S}$ can also take both positive and negative values resulting in a saddle stationary point. Thus, the underlying dynamical system would have both stable and unstable modes making it impossible for the algorithm in (4.6) with fixed sign step-size to converge. This is well known in the literature [3,14]. However, as will be shown next, this difficulty can be removed for our case by appropriately utilizing the sign of the update equation (remember that this saddle point is the only stationary point of the quadratic performance surface). The general idea is to use a vector step-size (one step-size per weight) having both *positive and negative values*. One unrealistic way (for an on-line algorithm) to achieve this goal is to estimate the eigenvalues of $\mathbf{R}-|\beta|\mathbf{S}$. Alternatively, we can derive the conditions on the step-size for guaranteed convergence. As before, we will define the error vector at time instant n as $\boldsymbol{\xi}(n)=\mathbf{w}_*-\mathbf{w}(n)$. The norm of the error vector at time instant n+1 is given by

$$\begin{aligned}\|\boldsymbol{\xi}(n+1)\|^2 = \|\boldsymbol{\xi}(n)\|^2 - 2\eta(n)\left[\boldsymbol{\xi}^T(n)e(n)\mathbf{x}(n)-|\beta|\boldsymbol{\xi}^T(n)\dot{e}(n)\dot{\mathbf{x}}(n)\right]+\\ \eta^2(n)\left\|e(n)\mathbf{x}(n)-|\beta|\dot{e}(n)\dot{\mathbf{x}}(n)\right\|^2\end{aligned} \tag{4.17}$$

Taking the expectations on both sides, we get

$$\begin{aligned}E\|\boldsymbol{\xi}(n+1)\|^2 = E\|\boldsymbol{\xi}(n)\|^2 - 2\eta(n)E\left[\boldsymbol{\xi}^T(n)e(n)\mathbf{x}(n)-|\beta|\boldsymbol{\xi}^T(n)\dot{e}(n)\dot{\mathbf{x}}(n)\right]+\\ \eta^2(n)E\left\|e(n)\mathbf{x}(n)-|\beta|\dot{e}(n)\dot{\mathbf{x}}(n)\right\|^2\end{aligned} \tag{4.18}$$

The mean of the error vector norm will monotonically decay to zero over time i.e., $E\|\boldsymbol{\xi}(n+1)\|^2 < E\|\boldsymbol{\xi}(n)\|^2$ if and only if the step-size satisfies the following inequality.

$$|\eta(n)| < \frac{2\left|E\left[\boldsymbol{\xi}^T(n)e(n)\mathbf{x}(n)-|\beta|\boldsymbol{\xi}^T(n)\dot{e}(n)\dot{\mathbf{x}}(n)\right]\right|}{E\left\|e(n)\mathbf{x}(n)-|\beta|\dot{e}(n)\dot{\mathbf{x}}(n)\right\|^2} \tag{4.19}$$

Let $\mathbf{x}(n) = \tilde{\mathbf{x}}(n) + \mathbf{v}(n)$ and $d(n) = \tilde{d}(n) + u(n)$, where $\tilde{\mathbf{x}}(n)$ and $\tilde{d}(n)$ be the clean input and desired data respectively. We will further assume that the input noise vector $\mathbf{v}(n)$ and the noise component in the desired signal $u(n)$ to be uncorrelated. Also the noise signals are assumed to be independent of the clean input and desired signals. Furthermore, the lag L is chosen to be more than *m*, the length of the filter under consideration. Since the noise is assumed to be purely white, $E[\mathbf{v}(n)\mathbf{v}^T(n-L)] = E[\mathbf{v}(n-L)\mathbf{v}^T(n)] = 0$ and $E[\mathbf{v}(n)\mathbf{v}^T(n)] = \mathbf{V}$. We have

$$\boldsymbol{\xi}^T(n)e(n)\mathbf{x}(n) = \left(\mathbf{w}_* - \mathbf{w}(n)\right)^T \left(\tilde{d}(n) + u(n) - \mathbf{w}^T(n)\tilde{\mathbf{x}}(n) - \mathbf{w}^T(n)\mathbf{v}(n)\right)\left(\tilde{\mathbf{x}}(n) + \mathbf{v}(n)\right) \quad (4.20)$$

Simplifying this further and taking the expectations, we get

$$\begin{aligned} E[\boldsymbol{\xi}^T(n)e(n)\mathbf{x}(n)] &= \mathrm{var}\left(\tilde{d}(n)\right) - 2\tilde{\mathbf{P}}^T\mathbf{w}(n) + \mathbf{w}^T(n)\tilde{\mathbf{R}}\mathbf{w}(n) \\ &\quad + \mathbf{w}^T(n)\mathbf{V}\mathbf{w}(n) - \mathbf{w}_*^T\mathbf{V}\mathbf{w}(n) \\ &= J_{MSE} - \mathbf{w}_*^T\mathbf{V}\mathbf{w}(n) \end{aligned} \quad (4.21)$$

where, $\tilde{\mathbf{R}} = E[\tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^T(n)]$, $\tilde{\mathbf{P}} = E[\tilde{\mathbf{x}}(n)\tilde{d}(n)]$ and

$$J_{MSE} = \mathbf{w}^T(n)\left(\tilde{\mathbf{R}} + \mathbf{V}\right)\mathbf{w}(n) + \mathrm{var}\left(\tilde{d}(n)\right) - 2\tilde{\mathbf{P}}^T\mathbf{w}(n) \quad (4.22)$$

Similarly, we have

$$\begin{aligned} \boldsymbol{\xi}^T(n)\dot{e}(n)\dot{\mathbf{x}}(n) &= (\mathbf{w}_* - \mathbf{w}(n))^T[\tilde{d}(n) + u(n) - \mathbf{w}^T(n)\left(\tilde{\mathbf{x}}(n) + \mathbf{v}(n)\right) \\ &\quad - \tilde{d}(n-L) + u(n-L) + \mathbf{w}^T(n)\left(\tilde{\mathbf{x}}(n-L) + \mathbf{v}(n-L)\right)] \\ &\quad \left(\tilde{\mathbf{x}}_k + \mathbf{v}_k - \tilde{\mathbf{x}}_{k-L} - \mathbf{v}_{k-L}\right) \end{aligned} \quad (4.23)$$

Evaluating the expectations on both sides of (4.23) and simplifying, we obtain

$$\begin{aligned} E\left[\boldsymbol{\xi}^T(n)\dot{e}(n)\dot{\mathbf{x}}(n)\right] &= \mathrm{var}\left(\tilde{d}(n) - \tilde{d}(n-L)\right) - 2\tilde{\mathbf{Q}}^T\mathbf{w}(n) \\ &\quad + \mathbf{w}^T(n)\tilde{\mathbf{S}}\mathbf{w}(n) + 2\mathbf{w}^T(n)\mathbf{V}\mathbf{w}(n) - 2\mathbf{w}_*^T(n)\mathbf{V}\mathbf{w}(n) \\ &= J_{ENT} - 2\mathbf{w}_*^T\mathbf{V}\mathbf{w}(n) \end{aligned} \quad (4.24)$$

where, we have used the definitions $\tilde{\mathbf{S}} = E[(\tilde{\mathbf{x}}(n) - \tilde{\mathbf{x}}(n-L))(\tilde{\mathbf{x}}(n) - \tilde{\mathbf{x}}(n-L))^T]$,

$\widetilde{\mathbf{Q}} = E[(\widetilde{\mathbf{x}}(n) - \widetilde{\mathbf{x}}(n-L))(\widetilde{d}(n) - \widetilde{d}(n-L))]$ and

$$J_{ENT} = \mathbf{w}^T(n)\left(\widetilde{\mathbf{S}} + 2\mathbf{V}\right)\mathbf{w}(n) + \mathrm{var}\left(\widetilde{d}(n) - \widetilde{d}(n-L)\right) - 2\widetilde{\mathbf{Q}}^T\mathbf{w}(n) \tag{4.25}$$

Using (4.21) and (4.24) in equation (4.19), we get an expression for the upper bound on the step-size as

$$|\eta(n)| < \frac{2\left|J_{MSE} - |\beta| J_{ENT} - \left(1 - 2|\beta|\right)\mathbf{w}_*^T\mathbf{V}\mathbf{w}(\mathrm{n})\right|}{E\left\|e(n)\mathbf{x}(n) - |\beta|\dot{e}(n)\dot{\mathbf{x}}(n)\right\|^2} \tag{4.26}$$

This expression is not usable in practice as an upper bound because it depends on the optimal weight vector. However, for $\beta = -0.5$, the upper bound on the step-size reduces to

$$|\eta(n)| < \frac{2\left|J_{MSE} - 0.5J_{ENT}\right|}{E\left\|e(n)\mathbf{x}(n) - 0.5\dot{e}(n)\dot{\mathbf{x}}(n)\right\|^2} \tag{4.27}$$

From (4.22) and (4.25), we know that J_{MSE} and J_{ENT} are positive quantities. However, $J_{MSE} - 0.5J_{ENT}$ can be negative. Also note that this upper bound is computed by evaluating the right hand side of (4.27) with the current weight vector $\mathbf{w}(n)$. Thus as expected, it is very clear that the step-size at the n^{th} iteration can take either positive or negative values based on $J_{MSE} - 0.5J_{ENT}$; therefore, $\mathrm{sgn}(\eta(n))$ must be the same as $\mathrm{sgn}(J_{MSE} - 0.5J_{ENT})$ evaluated at $\mathbf{w}(n)$. Intuitively speaking, the term $J_{MSE} - 0.5J_{ENT}$ is the EWC cost computed with the current weights $\mathbf{w}(n)$ and $\beta = -0.5$, which tells us where we are on the performance surface and the sign tells which way to go to reach the stationary point. It also means that the lower bound on the step-size is not positive as in traditional gradient algorithms. In general, if the step-size we choose satisfies (4.27), then, the mean error vector norm decreases asymptotically i.e., $E\left\|\boldsymbol{\xi}(n+1)\right\|^2 < E\left\|\boldsymbol{\xi}(n)\right\|^2$

and eventually becomes zero, which implies that $\lim_{n\to\infty} E[\mathbf{w}(n)] \to \mathbf{w}_*$. Thus the weight vector $E[\mathbf{w}(n)]$ converges asymptotically to $\mathbf{w}_*$, which is the only stationary point of the ODE in (4.15). We conclude that the knowledge of the eigenvalues is not needed to implement gradient descent in the EWC performance surface, but (4.27) is still not appropriate for a simple LMS type algorithm.

On-line Implementations of AEC-LMS for $\beta < 0$

As mentioned before, computing $J_{MSE} - 0.5J_{ENT}$ at the current weight vector would require reusing the entire past data at every iteration. As an alternative, we can extract the curvature at the operating point and include that information in the gradient algorithm. By doing so, we obtain the following stochastic algorithm.

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta \operatorname{sgn}\left(\mathbf{w}^T(n)\left[\mathbf{R}(n) - |\beta|\mathbf{S}(n)\right]\mathbf{w}(n)\right)\left(e(n)\mathbf{x}(n) - |\beta|\dot{e}(n)\dot{\mathbf{x}}(n)\right) \quad (4.28)$$

where, $\mathbf{R}(n)$ and $\mathbf{S}(n)$ are the estimates of $\mathbf{R}$ and $\mathbf{S}$ respectively at the n^{th} time instant.

Corollary: Given any quadratic surface $J(\mathbf{w})$, the following gradient algorithm converges to its stationary point.

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \eta \operatorname{sgn}(\mathbf{w}^T(n)\mathbf{H}\mathbf{w}(n)) \frac{\partial J}{\partial \mathbf{w}(n)} \quad (4.29)$$

Proof: Without loss of generality, suppose that we are given a quadratic surface of the form $J(\mathbf{w}) = \mathbf{w}^T\mathbf{H}\mathbf{w}$, where $\mathbf{H} \in \Re^{m\times m}$, and $\mathbf{w} \in \Re^{mx1}$. $\mathbf{H}$ is restricted to be symmetric; therefore, it is the Hessian matrix of this quadratic surface. The gradient of the performance surface with respect to the weights, evaluated at point $\mathbf{w}_0$ is $\frac{\partial J}{\partial \mathbf{w}_0} = 2\mathbf{H}\mathbf{w}_0$, and the stationary point of $J(\mathbf{w})$ is the origin. Since the performance surface is quadratic, any cross-section passing through the stationary point is a parabola. Consider the cross-

section of $J(\mathbf{w})$ along the line defined by the local gradient that passes through the point $\mathbf{w}_0$. In general, the Hessian matrix of this surface can be positive or negative definite; it might as well have mixed eigenvalues. The unique stationary point of $J(\mathbf{w})$, which makes its gradient zero, can be reached by moving along the direction of the local gradient. The important issue is the selection of the sign, i.e., whether to move along or against the gradient direction to reach the stationary point. The decision can be made by observing the local curvature of the cross-section of $J(\mathbf{w})$ along the gradient direction. The performance surface cross-section along the gradient direction at $\mathbf{w}_0$ is

$$J(\mathbf{w}_0 + 2\eta\mathbf{H}\mathbf{w}_0) = \mathbf{w}_0^T(I + 2\eta\mathbf{H})^T\,\mathbf{H}(I + 2\eta\mathbf{H})\mathbf{w}_0 = \mathbf{w}_0^T(\mathbf{H} + 4\eta\mathbf{H}^2 + 4\eta^2\mathbf{H}^3)\mathbf{w}_0 \qquad (4.30)$$

From this, we deduce that the local curvature of the parabolic cross-section at $\mathbf{w}_0$ is $4\mathbf{w}_0^T\mathbf{H}^3\mathbf{w}_0$. If the performance surface is locally convex, then this curvature is positive. If the performance surface is locally concave, this curvature is negative. Also, note that $\text{sgn}(4\mathbf{w}_0^T\mathbf{H}^3\mathbf{w}_0) = \text{sgn}(\mathbf{w}_0^T\mathbf{H}\mathbf{w}_0)$. Thus, the update equation with the curvature information in (4.30) converges to the stationary point of the quadratic cost function $J(\mathbf{w})$ irrespective of the nature of the stationary point.

From the above corollary and utilizing the fact that the matrix $\mathbf{R} - |\beta|\mathbf{S}$ is symmetric, we can conclude that the update equation in (4.29) asymptotically converges to the stationary point $\mathbf{w}_* = \left(\mathbf{R} - |\beta|\mathbf{S}\right)^{-1}\left(\mathbf{P} - |\beta|\mathbf{Q}\right)$. On the down side however, the update equation in (4.28) requires $O(m^2)$ computations, which makes the algorithm unwieldy for real-world applications. Also, we can use the REW algorithm instead, which has a similar complexity.

For an $O(m)$ algorithm, we have to go back to the update rule in (4.6). We will discuss only the simple case of $\beta = -0.5$, which turns out to be also the more useful. We propose to use an instantaneous estimate of the sign with the current weights given by

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)\,\mathrm{sgn}(e^2(n) - 0.5\dot{e}^2(n))[e(n)\mathbf{x}(n) - 0.5\dot{e}(n)\dot{\mathbf{x}}(n)] \tag{4.31}$$

where, where $\eta(n) > 0$ and is bound by (4.27). It is possible to make mistakes in the sign estimation when (4.31) is utilized, which will not affect the convergence in the mean, but will penalize the misadjustment.

Excess Error Correlation Bound for EWC-LMS

In the next theorem, we will show that the asymptotic excess error correlation at lags $L \geq m$ is always bounded from above and can be arbitrarily reduced by controlling the step-size.

Theorem 4.2: With $\beta = -1/2$, the steady state excess error autocorrelation at lag $L \geq m$, i.e., $|\rho_{\dot{e}}(L)|$ is always bound by

$$|\rho_{\dot{e}}(L)| \leq \frac{\eta}{2}[Tr(\mathbf{R}) + \sigma_v^2\mathbf{I}][E(e_a^2(k)) + \sigma_u^2 + \|\mathbf{w}\|_\infty^2 \sigma_v^2] \tag{4.32}$$

where $\mathbf{R} = E[\mathbf{x}_k\mathbf{x}_k^T]$, and $Tr(\bullet)$ denotes the matrix trace. The term $E(e_a^2(k))$ denotes the excess MSE which is $(\mathbf{w}_\infty - \mathbf{w}_T)^T\mathbf{R}(\mathbf{w}_\infty - \mathbf{w}_T)$. The noise variances in the input and desired signals are represented by σ_v^2 and σ_u^2 respectively. Note that the term $\|\mathbf{w}\|_\infty^2$ is always bound because of the step-size bound.

Proof: For convenience, we will adopt the subscript *k* to denote the time or iteration index. With this convention, the weight vector at the k^{th} iteration is denoted by $\mathbf{w}_k$ Further, the error vector (difference between the true vector $\mathbf{w}_T$ and the adaptive estimate

at time *k*) is denoted by $\hat{\boldsymbol{\varepsilon}}_k = \mathbf{w}_T - \mathbf{w}_k$. Throughout the rest of the proof, we will use the following notations: the noisy input vector $\hat{\mathbf{x}}_k$, the noise-free input vector $\mathbf{x}_k$, and the input noise vector $\mathbf{v}_k$ obey $\hat{\mathbf{x}}_k = \mathbf{x}_k + \mathbf{v}_k$; the noisy desired signal $\hat{d}_k$, the noise-free desired signal d_k, and noise u_k are related by $\hat{d}_k = d_k + u_k$.We will start from the equation describing the dynamics of the error vector norm given below.

$$\left\|\hat{\boldsymbol{\varepsilon}}_{k+1}\right\|^2 = \left\|\hat{\boldsymbol{\varepsilon}}_k\right\|^2 - 2\eta sign(\hat{e}_k^2 + \beta\hat{e}_k^2)\hat{\boldsymbol{\varepsilon}}_k^T(\hat{e}_k\hat{\mathbf{x}}_k + \beta\hat{e}_k\hat{\hat{\mathbf{x}}}_k) + \eta^2\left\|\hat{e}_k\hat{\mathbf{x}}_k + \beta\hat{e}_k\hat{\hat{\mathbf{x}}}_k\right\|^2 \tag{4.33}$$

In (4.33), we have assumed a constant step-size which satisfies the upper bound in (4.27). Letting $E\left\|\hat{\boldsymbol{\varepsilon}}_{k+1}\right\|^2 = E\left\|\hat{\boldsymbol{\varepsilon}}_k\right\|^2$ as $k \to \infty$, we see that

$$\frac{\eta}{2}E\left\|\hat{e}_k\hat{\mathbf{x}}_k + \beta\hat{e}_k\hat{\hat{\mathbf{x}}}_k\right\|^2 = E\left|\hat{\boldsymbol{\varepsilon}}_k^T(\hat{e}_k\hat{\mathbf{x}}_k + \beta\hat{e}_k\hat{\hat{\mathbf{x}}}_k)\right| \tag{4.34}$$

We now invoke the Jensen's inequality for convex functions [130] to reduce (4.34) further, yielding

$$\frac{\eta}{2}E\left\|\hat{e}_k\hat{\mathbf{x}}_k + \beta\hat{e}_k\hat{\hat{\mathbf{x}}}_k\right\|^2 \geq \left|E[\hat{\boldsymbol{\varepsilon}}_k^T(\hat{e}_k\hat{\mathbf{x}}_k + \beta\hat{e}_k\hat{\hat{\mathbf{x}}}_k)]\right| \tag{4.35}$$

The noisy error term is given by $\hat{e}_k = e_a(k) + u_k - \mathbf{w}^T\mathbf{v}_k$ where the excess error $e_a(k) = \boldsymbol{\varepsilon}^T\mathbf{x}_k$. Using the expressions $E[\hat{\boldsymbol{\varepsilon}}_k^T\hat{e}_k\hat{\mathbf{x}}_k] = E(e_k^2) - \mathbf{w}_*^T\mathbf{V}\mathbf{w}_k + \mathbf{w}_k^T\mathbf{V}\mathbf{w}_k$, $E[\hat{\boldsymbol{\varepsilon}}_k^T\hat{e}_k\hat{\hat{\mathbf{x}}}_k] = E(\dot{e}_k^2) - 2\mathbf{w}_*^T\mathbf{V}\mathbf{w}_k + 2\mathbf{w}_k^T\mathbf{V}\mathbf{w}_k$ and $\beta = -0.5$, we can immediately recognize that the RHS of (4.35) is simply the steady state excess error autocorrelation at lag $L \geq m$, i.e., $\left|\rho_{\hat{e}}(L)\right|$. In order to evaluate the LHS of (4.35), we will assume that the terms that $\left\|\hat{\mathbf{x}}_k\right\|^2$ and $\eta\hat{e}_k^2$ are uncorrelated in the steady state. Using, this assumption, we can write

$$\frac{\eta}{2}E\left\|\hat{e}_k\hat{\mathbf{x}}_k - 0.5\hat{\hat{e}}_k\hat{\mathbf{x}}_k\right\|^2 = \frac{\eta}{2}E(\hat{e}_k^2)[Tr(\mathbf{R}) + \sigma_v^2\mathbf{I}] \tag{4.36}$$

where, $E(\hat{e}_k^2) = E(e_a^2(k)) + \sigma_u^2 + \|\mathbf{w}\|_\infty^2\sigma_v^2$. Using (4.36) in equation (4.35), we get the inequality in (4.32).

This assumption (more relaxed than the independence assumptions [11,14]) is used in computing the steady state excess-MSE for stochastic LMS algorithm [131,132] and becomes more realistic for long filters. In the estimation of the excess MSE for the LMS algorithm, Price's theorem [133] for Gaussian random variables can be invoked to derive closed form expressions. However, even the Gaussianity assumption is questionable as discussed by Eweda [134] who proposed additional reasonable constraints on the noise pdf to overcome the Gaussianity and independence assumptions that lead to a more generic treatment for the sign-LMS algorithm. It is important to realize at this point that in the analysis presented here, no explicit Gaussianity assumptions have been made.

As a special case, consider $L = 0$ and noise free input. Then, (4.32) is true with the equality sign and also $|\rho_{\hat{e}}(L)|$ will be the same as $E(e_a^2(k))$ which is nothing but the excess MSE (as $k \to \infty$) of the LMS algorithm. In other words, (4.32) reduces to

$$E(e_a^2(k)) = \frac{\eta}{2}Tr(\mathbf{R})[E(e_a^2(k)) + \sigma_u^2] \tag{4.37}$$

From (4.37), the excess MSE for the LMS algorithm [14] can be deduced as

$$E(e_a^2(k)) = \frac{\eta\sigma_u^2 Tr(\mathbf{R})}{2 - \eta Tr(\mathbf{R})} \tag{4.38}$$

which will become to $\eta\sigma_u^2 Tr(\mathbf{R})/2$ for very small step-sizes. If the adaptive filter is long enough, the excess error $e_a(k)$ will be Gaussian and we can easily show that the excess

MSE is bounded by $Tr(\mathbf{R})E[\|\boldsymbol{\varepsilon}_0\|^2]/4$ where, $\boldsymbol{\varepsilon}_0$ denotes the error due to the initial condition [131].

Other Variants of the AEC-LMS Algorithms

It is easy to see that for convergence of the mean, the condition is $|I-\eta\lambda_k(\mathbf{R}+\beta\mathbf{S})|<1$ for all k, where $\lambda_k(\mathbf{R}+\beta\mathbf{S})$ denotes the k^{th} eigenvalue of the matrix $(\mathbf{R}+\beta\mathbf{S})$. This gives an upper bound on the step-size as $|\eta|<2/|\lambda_{\max}(\mathbf{R}+\beta\mathbf{S})|$. From the triangle inequality [8], $\|\mathbf{R}\|_2+|\beta|\|\mathbf{S}\|_2\le\sqrt{\lambda_{\max}(\mathbf{R})}+|\beta|\sqrt{\lambda_{\max}(\mathbf{S})}$ where, $\|\bullet\|_2$ denotes the matrix norm. Since, both **R** and **S** are positive-definite matrices, we can write

$$\|\mathbf{R}\|_2+|\beta|\|\mathbf{S}\|_2\le\sqrt{Tr(\mathbf{R})}+|\beta|\sqrt{Tr(\mathbf{S})}\le\sqrt{E\|\mathbf{x}(n)\|^2}+|\beta|\sqrt{E\|\dot{\mathbf{x}}(n)\|^2} \tag{4.39}$$

In a stochastic framework, we can include this in the AEC-LMS update equation to result in a step-size normalized EWC-LMS update rule given by

$$\mathbf{w}(n+1)=\mathbf{w}(n)+\frac{\eta sign(e^2(n)+\beta\dot{e}^2(n))(e(n)\mathbf{x}(n)+\beta\dot{e}(n)\dot{\mathbf{x}}(n))}{\left(\|\mathbf{x}(n)\|+|\beta|\|\dot{\mathbf{x}}(n)\|\right)^2} \tag{4.40}$$

Note that when $\beta=0$, (4.40) reduces to the well-known normalized LMS (NLMS) algorithm [14]. Alternatively, we can normalize by the norm squared of the gradient and this gives the following modified update rule.

$$\mathbf{w}(n+1)=\mathbf{w}(n)+2\frac{(e^2(n)+\beta\dot{e}^2(n))(e(n)\mathbf{x}(n)+\beta\dot{e}(n)\dot{\mathbf{x}}(n))}{\|e(n)\mathbf{x}(n)+\beta\dot{e}(n)\dot{\mathbf{x}}(n)\|^2+\delta} \tag{4.41}$$

The term δ, a small positive constant compensates for the numerical instabilities when the signal has zero power or when the error goes to zero, which can happen in the noiseless case even with finite number of samples. Once again, we would like to state that with $\beta=0$, (4.41) defaults to NLMS algorithm. However, the caveat is that, both (4.40) and

(4.41) do not satisfy the principle of minimum disturbance unlike the NLMS[3] [14]. But nevertheless, the algorithms in (4.40) and (4.41) can be used to provide faster convergence at the expense of increased misadjustment (in the error correlation sense) in the final solution.

AEC-LMS Algorithm with Multiple Lags

In the previous chapter, we discussed a recursive Newton type algorithm that included more than one lag in the cost function. With decreasing SNR at the input, the Hessian matrix $\mathbf{H} = \mathbf{R} + \beta\mathbf{S}$ is mostly determined by the noise covariance matrix. This can degrade the performance and we might be forced to use very small step-sizes (slow convergence) to achieve good results. One way of alleviating this problem is to incorporate multiple lags in the AEC cost function. The stochastic gradient AEC-LMS algorithm for the multiple lag case is simply given by

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \sum_{L=1}^{L_{\max}} \eta_L sign(e^2(n) + \beta\dot{e}_L^2(n))(e(n)\mathbf{x}(n) + \beta\dot{e}_L(n)\dot{\mathbf{x}}_L(n)) \qquad (4.42)$$

$L_{\max}$ is the total number of lags (constraints) used in the AEC cost function. The additional robustness of using multiple lags comes at an increase in the computational cost and in the case when the number of lags becomes equal to the length of the adaptive filter, the complexity will approach that of the recursive Newton type algorithms.

The stochastic AEC algorithms have linear complexity in comparison with the $O(\mathrm{N}^2)$ algorithm of the recursive Newton type algorithms discussed in the previous chapter. At the same time, since the algorithms are all based on the instantaneous

[3] The NLMS algorithm is also called the minimum norm update algorithm. It can be formulated as a constrained minimization problem wherein the actual cost function is the norm of the update, viz.,

$\|\mathbf{w}(n)-\mathbf{w}(n-1)\|^2$ and the constraint is the error e(n) with the weights $\mathbf{w}(n)$ must be zero.

gradients, these algorithms have better tracking abilities when compared with their Newton counterparts. Hence these algorithms can be expected to perform better in non-stationary conditions.

Simulation Results

Estimation of System Parameters in White Noise

The experimental setup is the same as the one used to test the REW algorithm. We varied the Signal-to-Noise-Ratio (SNR) between –10dB to +10dB and changed the number of filter parameters from 4 to 12. We set $\beta = -0.5$ and used the update equation in (4.31) for the EWC-LMS algorithm. A time varying step-size magnitude was chosen in accordance with the upper bound given by (4.27) without the expectation operators. This greatly reduces the computational burden but makes the algorithm noisier. However, since we are using 50,000 samples for estimating the parameters, we can expect the errors to average out over iterations. For the LMS algorithm, we chose the step-size that gave the least error in each trial. Totally 100 Monte Carlo trials were performed and histograms of normalized error vector norms were plotted. It is possible to use other statistical measures instead of the error norm, but this is sufficient to demonstrate the bias removal ability of EWC-LMS. For comparison purposes, we computed the solutions with LMS as well as the numerical TLS (regular TLS) methods. Figure 4-1 shows the error histograms for all the three methods. The inset plots in Figure 4-1 show the summary of the histograms for each method. EWC-LMS performs significantly better than LMS at low SNR values (-10dB and 0dB), while performing equally well for 10dB SNR. The input noise variances for -10dB, 0dB, and 10dB SNR values are 10, 1, and 0.1, respectively. Thus, we expect (and observe) TLS results to be worst for -10dB and best

for 10dB. As per theory, we observe that TLS performance drops when the noise variances are not the same in the input and desired signals.

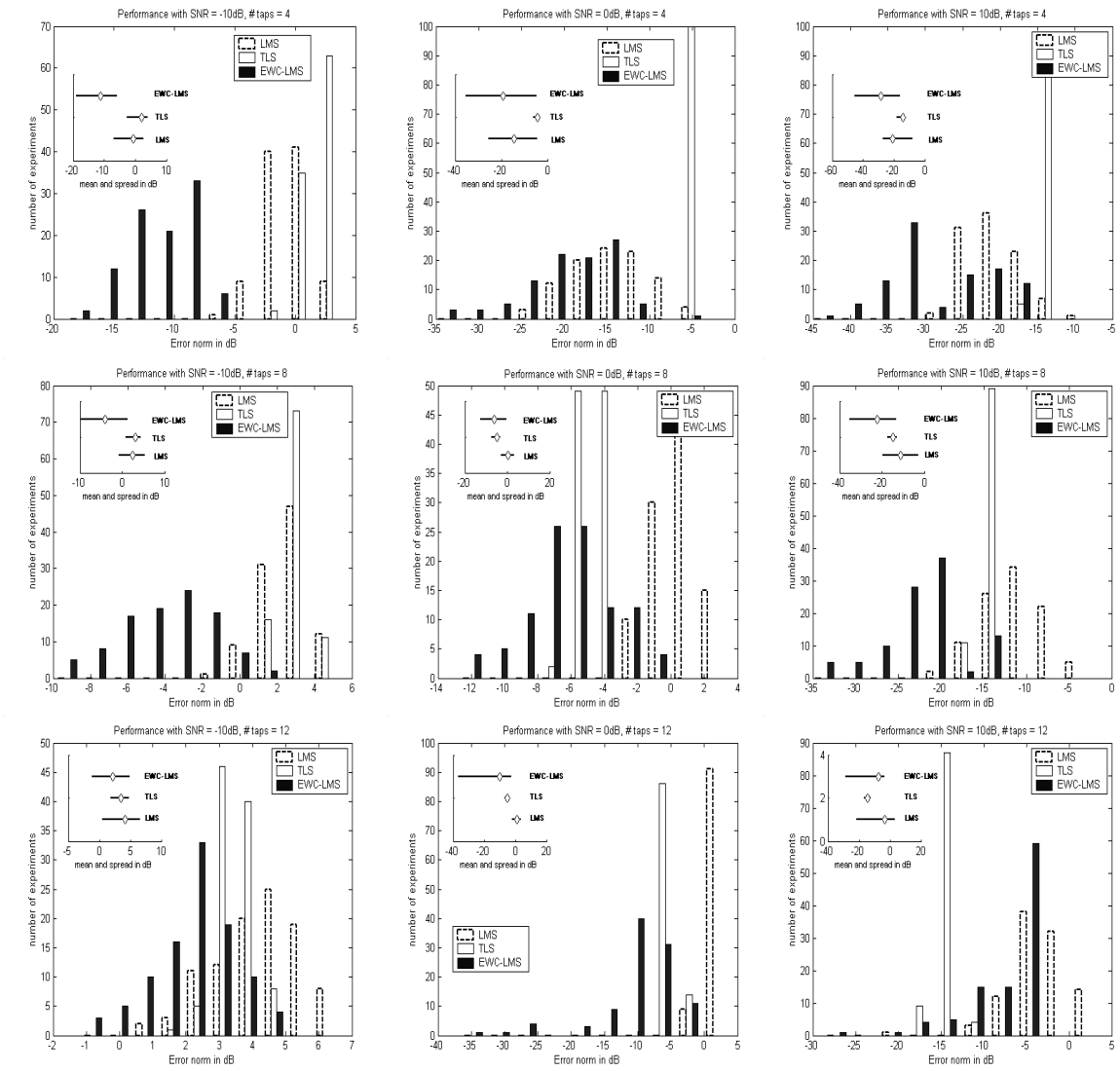


Figure 4-1. Histogram plots showing the error vector norm for EWC-LMS, LMS algorithms and the numerical TLS solution.

Figure 4-2 shows a sample comparison between the stochastic and the recursive algorithms for 0dB SNR and 4 filter taps. Interestingly, the performance of the EWC-LMS algorithm is better than the REW algorithm in the presence of noise. Similarly, the LMS algorithm is much better than the RLS algorithm. This tells us that the stochastic

algorithms presumably reject more noise than the fixed-point algorithms. Researchers have made this observation before, although no concrete arguments exist to account for the smartness of the adaptive algorithms [135]. Similar conclusions can be drawn in our case for EWC-LMS and REW.

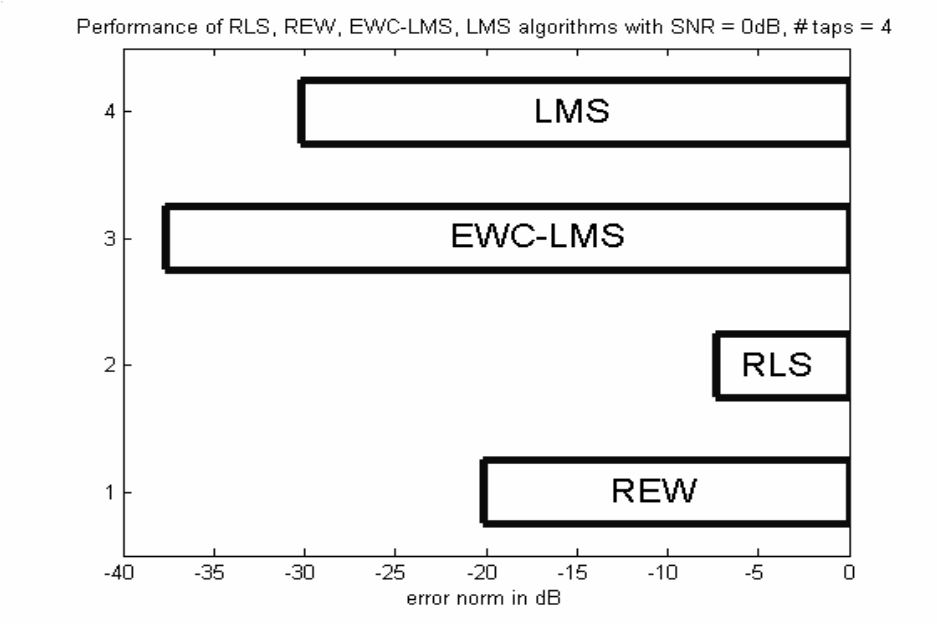


Figure 4-2. Comparison of stochastic versus recursive algorithms.

Weight Tracks and Convergence

The steady state performance of a stochastic gradient algorithm is a matter of great importance. We will now experimentally verify the steady state behavior of the EWC-LMS algorithm. The SNR of the input signal is set to 10dB and the number of filter taps is fixed to two for display convenience. Figure 4-3 shows the contour plot of the EWC cost function with noisy input data. Clearly, the Hessian of this performance surface has both positive and negative eigenvalues thus making the stationary point an undesirable saddle point. On the same plot, we have shown the weight tracks of the EWC-LMS algorithm with $\beta = -0.5$. Also, we used a fixed value of 0.001 for the step-size. From

the figure, it is clear that the EWC-LMS algorithm converges stably to the saddle point solution, which is theoretically unstable when a single sign step-size is used. Notice that due to the constant step-size, there is misadjustment in the final solution. In Figure 4-4, we show the individual weight tracks for the EWC-LMS algorithm. The weights converge to the vicinity of the true filter parameters, which are -0.2 and 0.5 respectively within 1000 samples.

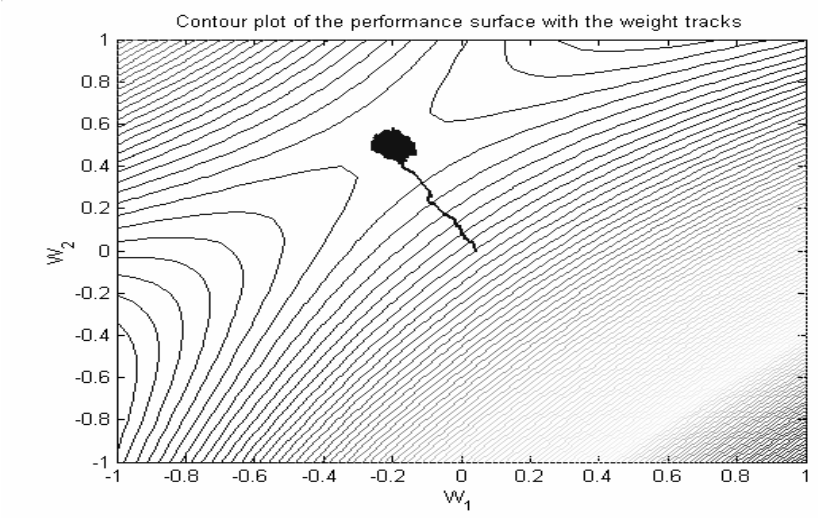


Figure 4-3. Contour plots with the weight tracks showing convergence to saddle point.

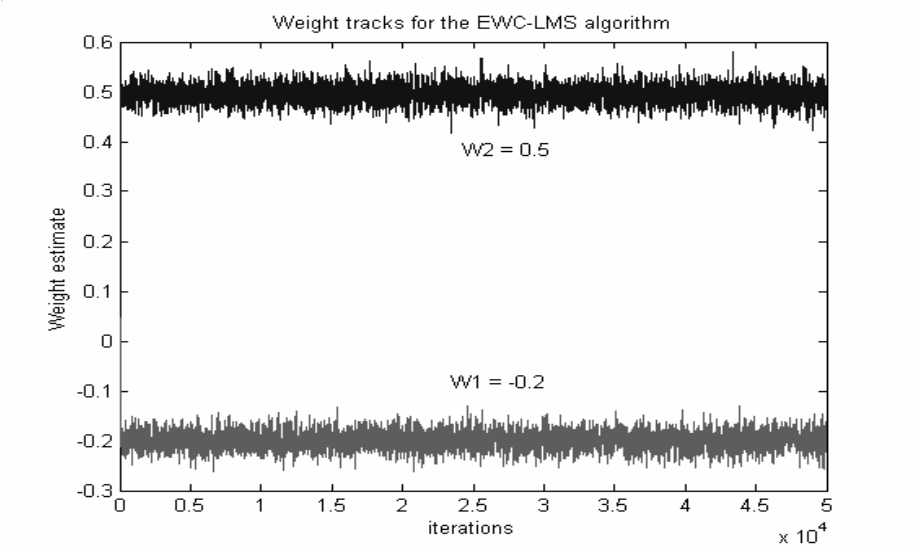


Figure 4-4. Weight tracks for the stochastic algorithm.

In order to see if the algorithm converges to the saddle point solution in a robust manner, we ran the same experiment using different initial conditions on the contours. Figure 4-5 shows a few plots of the weight tracks originating from different initial values over the contours of the performance surface. In every case, the algorithm converged to

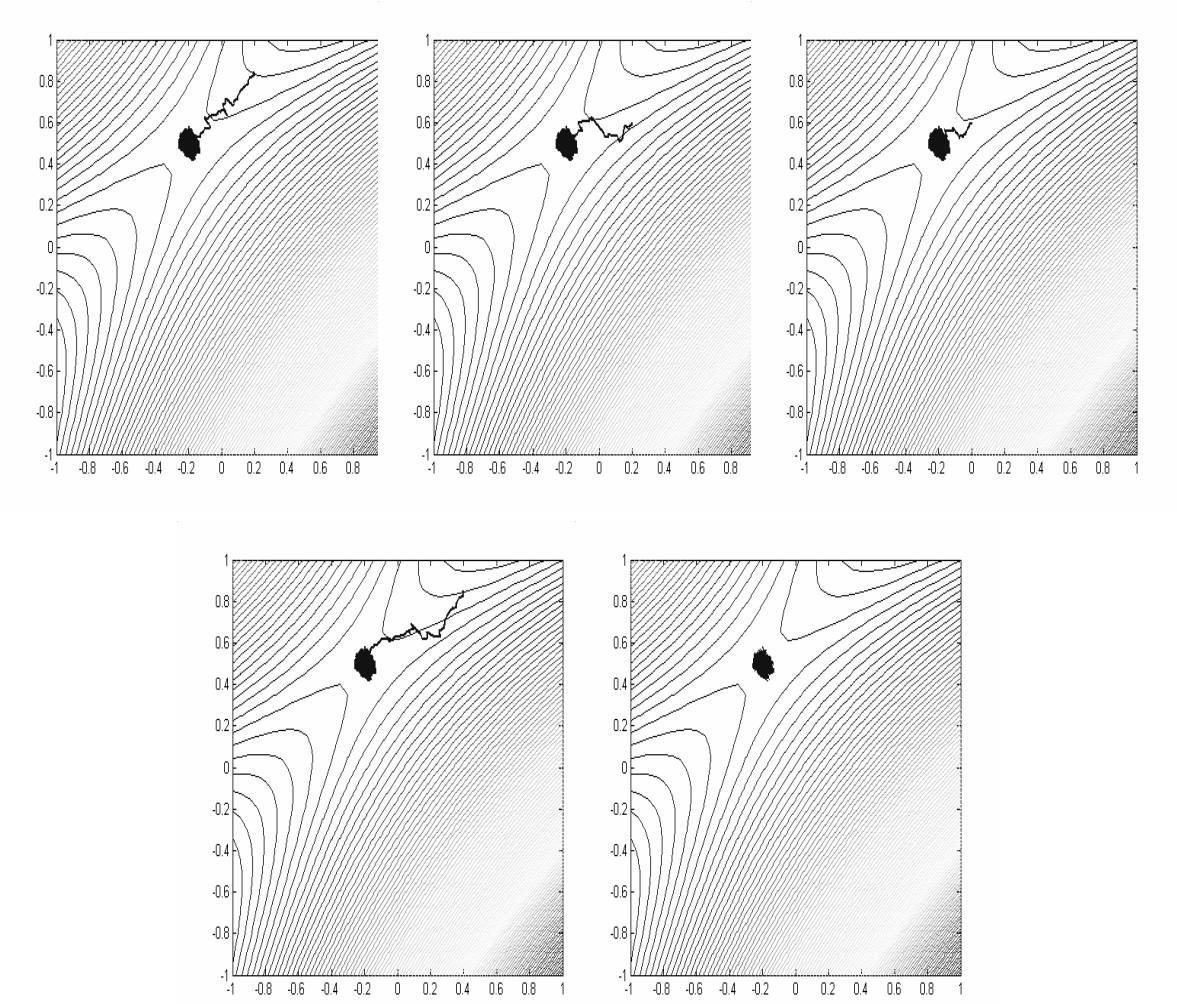


Figure 4-5. Contour plot with weight tracks for different initial values for the weights.

the saddle point in a stable manner. Note that the misadjustment in each case is almost the same. Finally, to quantify the effect of reducing the SNR, we repeated the experiment with 0dB SNR. Figure 4-6 (left) shows the weight tracks over the contour and we can see that the misadjustment has increased owing to decrease in the SNR. This is a typical

phenomenon observed with most of the stochastic gradient algorithms. However, the misadjustment is proportional to the step-size. Therefore, by using smaller step-sizes, the misadjustment can be controlled to be within acceptable values. The drawback is slow convergence to the optimal solution. Figure 4-6 (right) shows the weight tracks when the algorithm is used without the *sign* information for the step-size. Note that convergence is

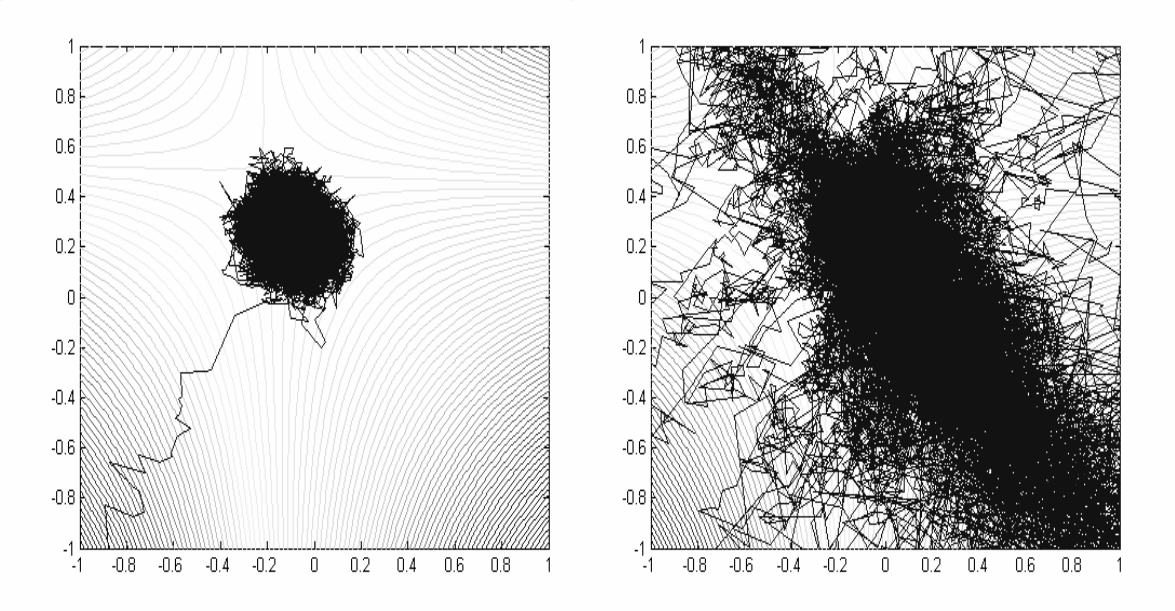


Figure 4-6. Contour plot with weight tracks for EWC-LMS algorithm with sign information (left) and without sign information (right).

not achieved in this case which substantiates our previous argument that a fixed sign step-size will never converge to a saddle point. To further substantiate this fact, we removed the noise from the input and ran the EWC-LMS algorithm with and without the *sign* term. Figure 4-7 (left) shows the noise-free EWC performance surface and Figure 4-7 (right) shows the weight tracks (with and without the *sign* information) on the contours. Clearly, the weights do not converge to the desired saddle point even in the absence of noise. On

the other hand, using the sign information leads the weights to the saddle point in a stable manner. Since this is the noise-free case, the final misadjustment becomes zero.

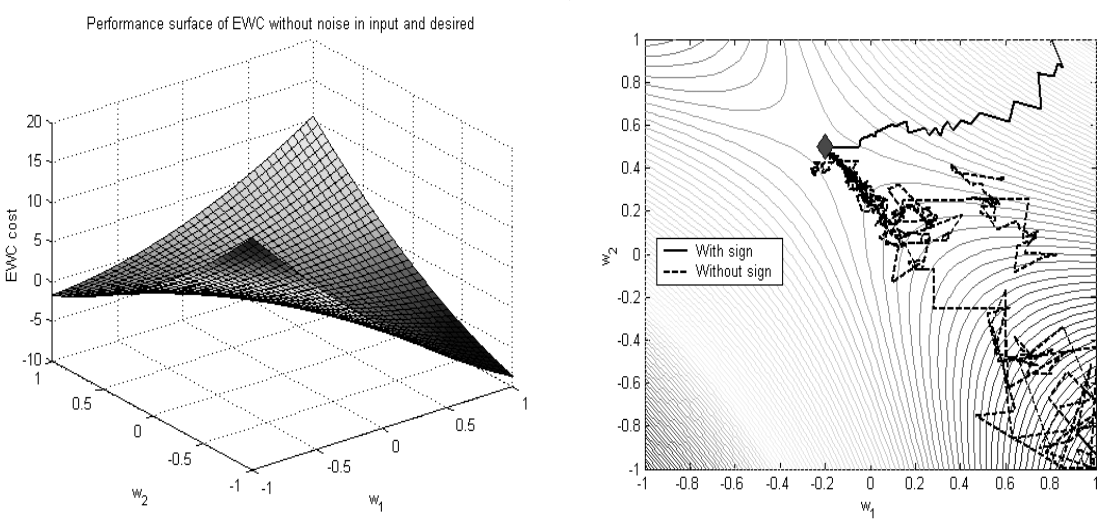


Figure 4-7. EWC performance surface (left) and weight tracks for the noise-free case with and without sign information (right).

Inverse Modeling and Controller Design Using EWC

System identification is the first step in the design of an inverse controller. Specifically, we wish to design a system that controls the plant to produce a predefined output. Figure 4-8 shows a block diagram of model reference inverse control [136]. In this case, the adaptive controller is designed so that the controller-plant pair would track the response generated by the reference model for any given input (command). Clearly, we require the plant parameters (which are typically unknown) to devise the controller. Once we have a model for the plant, the controller can be easily designed using conventional MSE minimization techniques. In this example, we will assume that the plant is an all-pole system with transfer function $P(z) = 1/(1 + 0.8z^{-1} - 0.5z^{-2} - 0.3z^{-3})$. The reference model is chosen to be an FIR filter with 5 taps. The block diagram for the plant identification is shown in Figure 4-9. Notice that the output of the plant is corrupted

with additive white noise due to measurement errors. The SNR at the plant output was set to 0dB. We then ran the EWC-LMS and LMS algorithms to estimate the model parameters given noisy input and desired signals. The model parameters thus obtained are used to derive the controller (see Figure 4-8) using standard backpropagation of error. We then tested the adaptive controller-plant pair for trajectory tracking by feeding the controller-plant pair with a non-linear time series and observing the responses. Ideally, the controller-plant pair must follow the trajectory generated by the reference model. Figure 4-10 (top) shows the tracking results for both controller-plant pairs along with the reference output. Figure 4-10 (bottom) shows a histogram of the tracking errors. Note

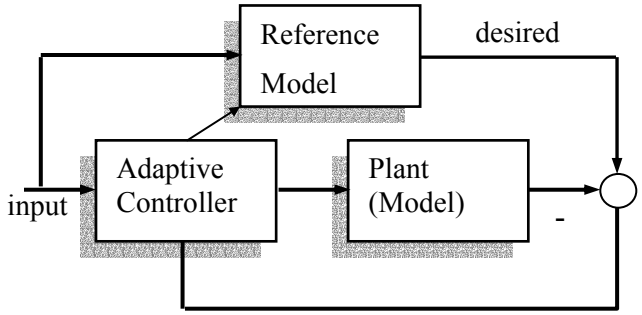


Figure 4-8. Block diagram for model reference inverse control.

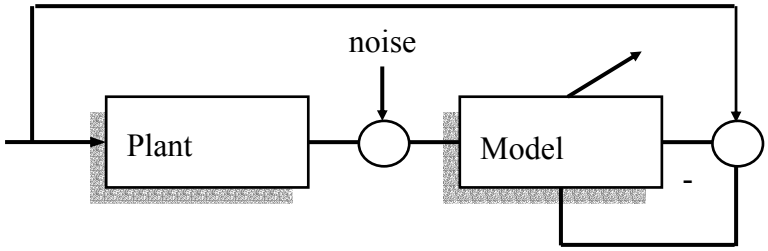


Figure 4-9. Block diagram for inverse modeling.

that the errors with EWC-LMS controller are all concentrated around zero, which is desirable. In contrast, the errors produced with the MSE based controller are significant

and this can be worse if the SNR levels drop further. Figure 4-11 shows the magnitude and phase responses of the reference models along with the generated controller-model

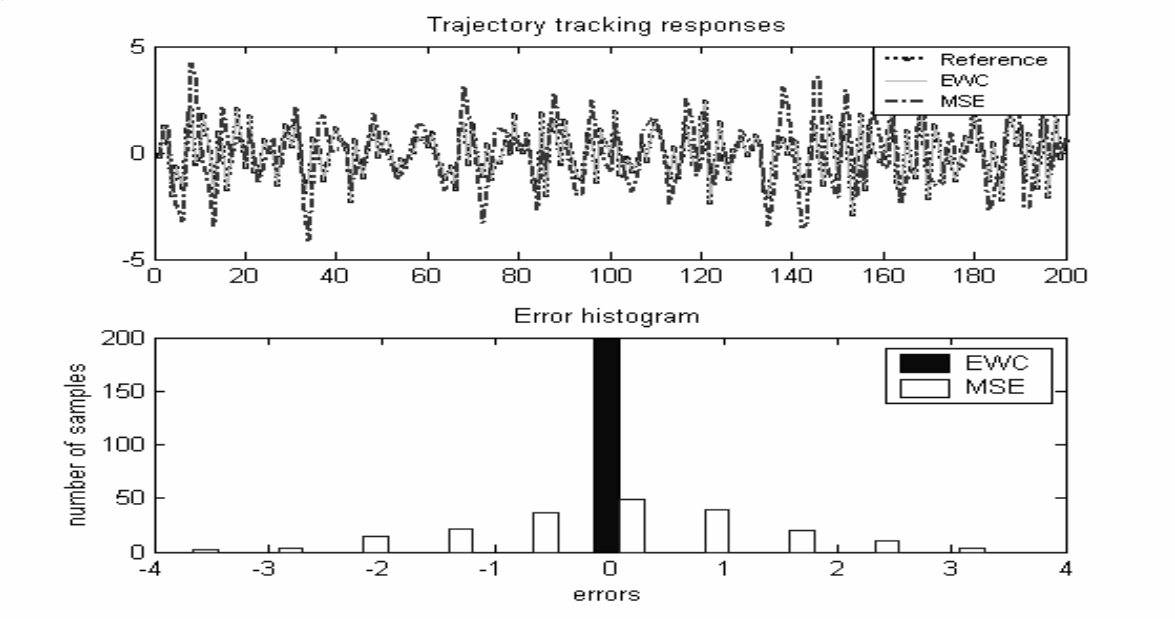


Figure 4-10. Plot of tracking results and error histograms.

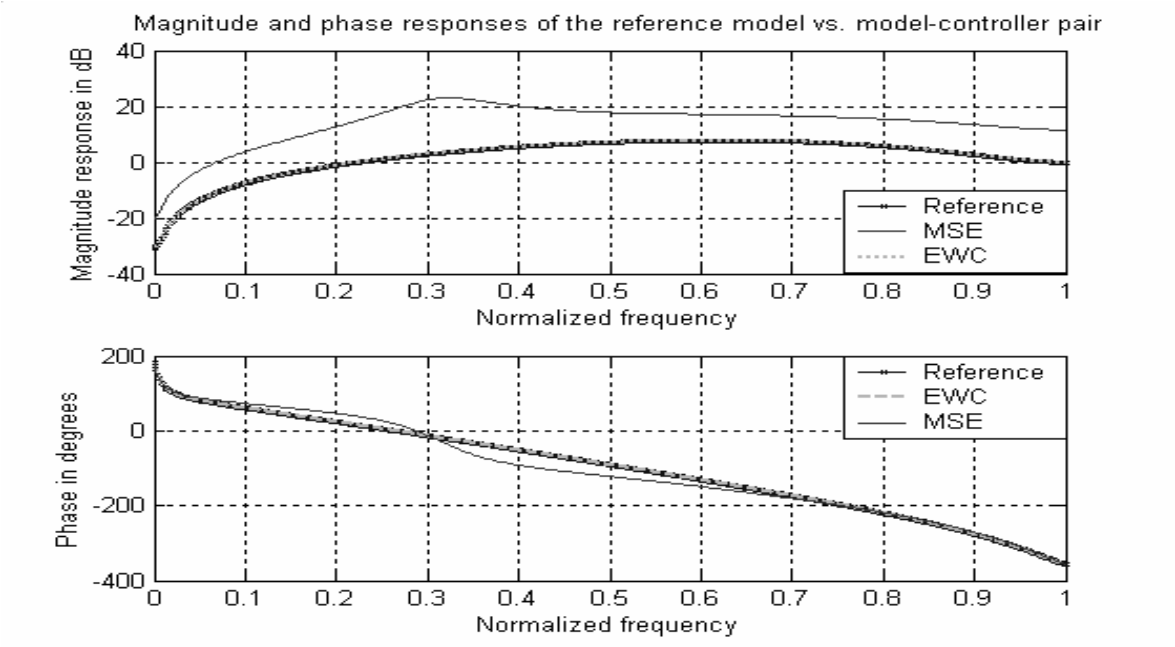


Figure 4-11. Magnitude and phase responses of the reference model and designed model-controller pairs.

pairs. Note that, the EWC controller-model pair matches very closely with the desired transfer function, whereas MSE controller-model pair produces a significantly different transfer function. This clearly demonstrates the advantages offered by EWC.

More details on the applications of EWC-LMS in system identification and controller design problems can be found in [137-139].

Summary

In this chapter, we proposed online sample-by-sample stochastic gradient algorithms for estimating the optimal AEC solution. The detailed derivations of the update rules were presented and the convergence was proved rigorously using stochastic approximation theory. We also derived the step-size upper bounds for convergence with probability one. Further, the theoretical upper bound on the excess error correlation in the case of EWC-LMS was derived. The AEC stochastic algorithms include the LMS algorithm for MSE as a special case. Owing to the complexities of the EWC performance surface (see Chapter 2), additional information like the *sign* of the instantaneous cost is required for guaranteed convergence to the unique optimal AEC solution. In this context, the AEC optimization problem can be pursued as a *root-finding* problem and the popular Robbins-Munro method [140] can be adopted to solve for the optimal solution. We have not explored this method yet for the AEC criterion.

We also presented several variants of the AEC-LMS algorithm. As a special case, the normalized AEC-LMS algorithm in equation (4.40) reduces to the well-known NLMS algorithm for MSE. The gradient normalized AEC-LMS algorithm in equation (4.41) has shown better performance over the simple AEC-LMS algorithm in our simulation studies.

We then presented simulation results to show the noise rejection capability of the EWC-LMS algorithm. Experiments were also conducted to verify some of the properties

of the proposed gradient algorithms. In particular, we observed the weight tracks and the verified that the algorithm converges in a stable manner to even saddle stationary points. This is achieved mainly by utilizing the *sign* information in the gradient update. We also showed the amount of misadjustment can be controlled by the step-size parameter. This is in conformance with the general theory behind stochastic gradient algorithms.

Lastly, we demonstrated the application of EWC in the design of a model-reference inverse controller. We compared the performance of the EWC controller with the MSE derived controller and verified the superiority of the former.

CHAPTER 5
LINEAR PARAMETER ESTIMATION IN CORRELATED NOISE

Introduction

In the previous chapters we discussed a new criterion titled augmented error criterion (AEC) that can potentially replace the popular MSE criterion. In fact we showed that a special case of the AEC called the error whitening criterion (EWC) can solve the problem of estimating the parameters of a linear system in the presence of input noise. We showed extensive simulation results with different EWC adaptation algorithms that proved beyond doubt, the usefulness of this criterion in solving system identification and controller design problems.

Two crucial assumptions were made in the theory behind the error whitening criterion. Firstly, we assumed that the input noise is uncorrelated with itself or is *white*. Although, in most problems, we assume that the noise is white, this assumption can be certainly restrictive in many applications. From the theory we discussed in the previous chapters, it is easy to conclude that EWC fails to remove the bias in the parameter estimates when the noise is correlated or *colored*.

Secondly, we assumed full knowledge of the model order of the unknown system. This is not just native to the proposed method as most of the competing methods including Total Least-Squares (TLS) assume exact model order. To the best of our knowledge, there is no existing solution to the problem of system identification in the presence of input noise in cases when the model order is unknown. However, till this point, we have not dealt with the implications of using the proposed EWC when the

model order is not known. We will address this important issue in the next chapter. In this chapter, we will focus on solving the problem of linear parameter estimation in the presence of correlated noise in the input and desired data. Most of the material covered in this chapter can be found in [141].

Existing Solutions

This problem of parameter estimation with noisy data has been a well researched topic although the solutions are not satisfactory. First and foremost, the MSE criterion does not provide accurate estimates in the presence of correlated noise. The regular TLS method assumes i.i.d. noise and hence fails when the noise is correlated. Further, there is the additional restriction of having equal noise variances in the input and desired data. The extension of TLS called the Generalized TLS (GTLS) discussed in Chapter 1 can be used for the correlated noise case. However, a major drawback is that we require an exact knowledge of the noise covariance matrix. Regalia gave a conceptual treatment for the IIR filter estimation based on equation-error techniques with the monic constraint replaced by a unit-norm constraint [92]. Douglas *et al*. extended the work to colored noise case in [142]. However, these methods require estimation of the noise covariances from the data, which is again not feasible. The Instrumental Variables (IV) method is traditionally limited to *white* noise. Extensions to the *colored* noise case are usually accomplished by introducing whitening filters [93]. The usual approach is to assume that the correlated noise is produced by filtering a *white* process by an AR model with known order. Thus, the problem then reduces to finding more parameters (the AR model parameters plus the actual system parameters) than usual, assuming white noise in the input. However, there are many loopholes in this technique. Most importantly, it is impossible to figure out the exact order of the noise AR process.

Criterion for Estimating the Parameters in Correlated Noise

Our goal is to propose a method to estimate the unknown system parameters without computing the input noise covariance matrices. We will approach this problem in two steps. In the first step, we will allow the noise in the input to be correlated, but we will restrict the noise on the desired signal to be white. The reasoning behind this step will be clear later. Once the algorithm is presented, we will introduce modifications in the algorithm that would enable us to remove the whiteness restriction on the noise in the desired signal. Further, in this research, we have restricted the unknown linear systems to be FIR filters. Generalizations to the IIR filter estimation and the associated stability issues are topics for further research.

A traditional setting of the system identification problem is shown in Figure 5-1. Suppose noisy training data pair $(\hat{\mathbf{x}}_k, \hat{d}_k)$ is provided, where $\hat{\mathbf{x}}_k \in \Re^N = \mathbf{x}_k + \mathbf{v}_k$ and $\hat{d}_k \in \Re^1 = d_k + u_k$ with $\mathbf{x}_k$ as the noise-free input vector at discrete time index *k*, $\mathbf{v}_k$, the additive noise vector with arbitrary covariance $\mathbf{V} = E[\mathbf{v}_k \mathbf{v}_k^T]$ on the input, d_k being the noise-free desired signal and u_k being the additive white noise added to the desired signal. We further assume that the noises $\mathbf{v}_k$ and u_k are independent from the data pair and also independent from each other. Let the weight vector (filter) that generated the noise-free data pair $(\mathbf{x}_k, d_k)$ be $\mathbf{w}_T$, of dimension *N*. We will assume that the length of $\mathbf{w}$, the estimated weight vector is *N* (sufficient order case). Then, the error sample $\hat{e}_k$ is simply given by $\hat{e}_k = \hat{d}_k - \mathbf{w}^T \hat{\mathbf{x}}_k$. Consider the cost function in equation (5.1).

$$J(\mathbf{w}) = \sum_{\Delta=1}^{N} \left| E[\hat{e}_k \hat{d}_{k-\Delta} + \hat{e}_{k-\Delta} \hat{d}_k] \right| \tag{5.1}$$

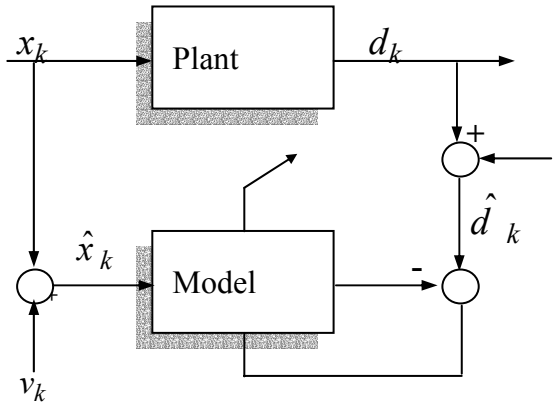


Figure 5-1. System identification block diagram showing data signals and noise.

Consider a single term in the above equation. It is easy to see that the cross products $E[\hat{e}_k \hat{d}_{k-\Delta}]$ and $E[\hat{e}_{k-\Delta} \hat{d}_k]$ are given by

$$\begin{aligned} E[\hat{e}_k \hat{d}_{k-\Delta}] &= \mathbf{w}_T^T E[\mathbf{x}_k \mathbf{x}_{k-\Delta}^T]\mathbf{w}_T - \mathbf{w}_T^T E[\mathbf{x}_k \mathbf{x}_{k-\Delta}^T]\mathbf{w} + E[u_k u_{k-\Delta}] \\ E[\hat{e}_{k-\Delta} \hat{d}_k] &= \mathbf{w}_T^T E[\mathbf{x}_{k-\Delta} \mathbf{x}_k^T]\mathbf{w}_T - \mathbf{w}_T^T E[\mathbf{x}_{k-\Delta} \mathbf{x}_k^T]\mathbf{w} + E[u_{k-\Delta} u_k] \end{aligned} \tag{5.2}$$

If we assume that the noise u_k is white, then $E[u_k u_{k-\Delta}] = 0$, and (5.2) reduces to functions of only the clean input and the weights. The input noise never multiplies itself; hence it gets eliminated. Further, the cost function in (5.1) simplifies to

$$J(\mathbf{w}) = \sum_{\Delta=1}^{N} \left| \mathbf{w}_T^T \mathbf{R}_\Delta \mathbf{w}_T - \mathbf{w}_T^T \mathbf{R}_\Delta \mathbf{w} \right| \tag{5.3}$$

where, the matrix $\mathbf{R}_\Delta$ is,

$$\mathbf{R}_\Delta = E[\mathbf{x}_k \mathbf{x}_{k-\Delta}^T + \mathbf{x}_{k-\Delta} \mathbf{x}_k^T] \tag{5.4}$$

The matrix $\mathbf{R}_\Delta$ is symmetric, but indefinite and hence can have mixed eigenvalues. Also, observe that the cost function in (5.3) is *linear* in the weights **w**. If, for instance, we had a single term in the summation, and we force $J(\mathbf{w}) = 0$, then it is easy to see that one of the solutions for **w** will be the true parameter vector $\mathbf{w}_T$. However, when the number of

terms in the summation becomes equal to the length of our estimated filter, there is always a unique solution for **w**, which will be the true vector $\mathbf{w}_T$.

Lemma 1: For suitable choices of lags, there is a unique solution $\mathbf{w}_*$ for the equation $J(\mathbf{w}_*) = 0$ and $\mathbf{w}_* = \mathbf{w}_T$.

Proof: For $J(\mathbf{w}) = 0$, $\mathbf{w}_T^T \mathbf{R}_\Delta \mathbf{w}_T - \mathbf{w}_T^T \mathbf{R}_\Delta \mathbf{w}$ must be zero for all selected Δ. For simplicity assume $\Delta = 1,..., N$. Therefore, we have *N* linear equations in **w** given by, $[\mathbf{w}_T^T \mathbf{R}_\Delta]\mathbf{w} = \mathbf{w}_T^T \mathbf{R}_\Delta \mathbf{w}_T$. This system of equations can be compactly written as

$$\begin{bmatrix} \mathbf{w}_T^T \mathbf{R}_1 \\ \mathbf{w}_T^T \mathbf{R}_2 \\ .... \\ \mathbf{w}_T^T \mathbf{R}_N \end{bmatrix} \mathbf{w} = 2 \begin{bmatrix} E[d_k d_{k-1}] \\ E[d_k d_{k-2}] \\ .... \\ E[d_k d_{k-N}] \end{bmatrix} \tag{5.5}$$

If the rows of the composite matrix on the left of **w** in (5.5) are linearly independent (full-rank matrix), then there is a unique inverse and hence $J(\mathbf{w}) = 0$ has a unique solution. We will prove that this unique solution has to be $\mathbf{w}_T$ by contradiction. Let the true solution be $\mathbf{w}_* = \mathbf{w}_T + \boldsymbol{\varepsilon}$. Then, $J(\mathbf{w}_*) = 0$ implies $\mathbf{w}_T^T \mathbf{R}_\Delta \boldsymbol{\varepsilon} = 0$ for all Δ which is possible only when $\boldsymbol{\varepsilon} = \mathbf{0}$ and this completes the proof.

Note that each term inside the summation of equation (5.1) can be perceived as a constraint on the cross correlation between the desired response and the error signal. By forcing these sums of cross correlations at *N* different lags to simultaneously approach zero, we can obtain an unbiased estimate of the true filter.

The optimal solution for the proposed criterion in terms of the noisy input and the desired responses is given in (5.6). Each row of the composite matrix can be estimated using simple correlators having linear complexity. Also, a recursive relationship for the

$$\mathbf{w}_* = 2\begin{bmatrix} E[\hat{d}_k\hat{\mathbf{x}}_{k-1}^T + \hat{d}_{k-1}\hat{\mathbf{x}}_k^T] \\ E[\hat{d}_k\hat{\mathbf{x}}_{k-2}^T + \hat{d}_{k-2}\hat{\mathbf{x}}_k^T] \\ .... \\ E[\hat{d}_k\hat{\mathbf{x}}_{k-N}^T + \hat{d}_{k-N}\hat{\mathbf{x}}_k^T] \end{bmatrix}^{-1}\begin{bmatrix} E[\hat{d}_k\hat{d}_{k-1}] \\ E[\hat{d}_k\hat{d}_{k-2}] \\ .... \\ E[\hat{d}_k\hat{d}_{k-N}] \end{bmatrix} \tag{5.6}$$

evolution of this matrix over iterations can be easily derived. However, this recursion does not involve simple reduced rank updates and hence it is not possible to use the convenient matrix inversion lemma [8] efficiently to reduce the complexity of matrix inversion. The overall complexity of the recursive solution in equation (5.6) is $O(\mathrm{N}^3)$. This motivates the development of a low cost stochastic algorithm to compute and track the optimal solution given by equation (5.6). The derivation of the stochastic gradient algorithm is similar to that of the AEC-LMS algorithm.

Stochastic Gradient Algorithm and Analysis

Taking the expectation operator out of the cost function in (5.1), we obtain an instantaneous cost given by

$$J(\mathbf{w}_k) = \sum_{\Delta=1}^{N}\left|\hat{e}_k\hat{d}_{k-\Delta} + \hat{e}_{k-\Delta}\hat{d}_k\right| \tag{5.7}$$

Again, we want to find the minimum of this cost function. Notice that the direction of the stochastic gradient in (5.7) depends on the instantaneous cost itself. The resulting weight update equation is given by

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \eta\sum_{\Delta=1}^{N} sign(\hat{e}_k\hat{d}_{k-\Delta} + \hat{e}_{k-\Delta}\hat{d}_k)(\hat{\mathbf{x}}_k\hat{d}_{k-\Delta} + \hat{\mathbf{x}}_{k-\Delta}\hat{d}_k) \tag{5.8}$$

where, $\eta > 0$ is a small step-size. The step-size has been chosen to be a constant in the above update equation. However, we can also have a time-varying step-size as before. Owing to the presence of multiple terms (constraints) in the gradient, the complexity of the update is $O(N^2)$ which is higher than that of the regular LMS type stochastic

updates. However, the complexity is lower than that of the recursive solution given by (5.7). We will now briefly discuss the convergence of this stochastic gradient algorithm to the optimal solution both in the noisy as well as noise-free scenarios.

Theorem 5.1: In the noise-free case, (5.8) converges to the stationary point $\mathbf{w}_* = \mathbf{w}_T$ provided that the step size satisfies the following inequality at every iteration.

$$0 < \eta < \frac{2J(\mathbf{w}_k)}{\left\|\nabla J(\mathbf{w}_k)\right\|^2} \tag{5.9}$$

Proof: It is obvious from the previous discussions that the cost function in (5.7) has a single stationary point $\mathbf{w}_* = \mathbf{w}_T$. The weight update becomes zero only when the cost goes to zero thereby zeroing the gradient. Consider the weight error vector defined as $\boldsymbol{\varepsilon}_k = \mathbf{w}_* - \mathbf{w}_k$. From (5.8), we get

$$\boldsymbol{\varepsilon}_{k+1} = \boldsymbol{\varepsilon}_k - \eta \sum_{\Delta=1}^{N} sign(e_k d_{k-\Delta} + e_{k-\Delta} d_k)(\mathbf{x}_k d_{k-\Delta} + \mathbf{x}_{k-\Delta} d_k) \tag{5.10}$$

Taking the norm of this error vector on both sides gives

$$\left\|\boldsymbol{\varepsilon}_{k+1}\right\|^2 = \left\|\boldsymbol{\varepsilon}_k\right\|^2 - 2\eta \sum_{\Delta=1}^{N} sign(e_k d_{k-\Delta} + e_{k-\Delta} d_k)\boldsymbol{\varepsilon}_k^T(\mathbf{x}_k d_{k-\Delta} + \mathbf{x}_{k-\Delta} d_k) + \eta^2 \left\|\nabla J(\mathbf{w}_k)\right\|^2 \tag{5.11}$$

Observe that in the noiseless case, $\boldsymbol{\varepsilon}_k^T \mathbf{x}_k = e_k$ and $\boldsymbol{\varepsilon}_k^T \mathbf{x}_{k-\Delta} = e_{k-\Delta}$. Hence (5.11) can be simplified to,

$$\left\|\boldsymbol{\varepsilon}_{k+1}\right\|^2 = \left\|\boldsymbol{\varepsilon}_k\right\|^2 - 2\eta \sum_{\Delta=1}^{N} \left|(e_k d_{k-\Delta} + e_{k-\Delta} d_k)\right| + \eta^2 \left\|\nabla J(\mathbf{w}_k)\right\|^2 \tag{5.12}$$

If we allow the error vector norm to decay asymptotically by forcing $\left\|\boldsymbol{\varepsilon}_{k+1}\right\|^2 < \left\|\boldsymbol{\varepsilon}_k\right\|^2$, we obtain the bound in (5.9). The error vector will eventually converge to zero by design, and since the gradient becomes null at the true solution: $\lim_{k\to\infty} \left\|\boldsymbol{\varepsilon}_k\right\|^2 \to 0$, and hence

$\lim_{k\to\infty} \mathbf{w}_k \to \mathbf{w}_* = \mathbf{w}_T$. This completes the proof.

Theorem 5.2: In the noisy data case, the stochastic algorithm in (5.8) converges to the stationary point $\mathbf{w}_* = \mathbf{w}_T$ in the mean provided that the step size is bound by the inequality given by

$$\eta < \frac{2\sum_{\Delta=1}^{N} \left| E[\hat{e}_k \hat{d}_{k-\Delta} + \hat{e}_{k-\Delta} \hat{d}_k] \right|}{E\left\| \nabla J(\mathbf{w}_k) \right\|^2} \tag{5.13}$$

Proof: Again, the facts about the uniqueness of the stationary point and it being equal to the true filter hold even for the noisy data case. The convergence to this stationary point in a stable manner will be proved in this theorem. Following the same steps as in the proof of the previous lemma, the dynamics of the error vector norm can be determined by the difference equation,

$$\left\| \boldsymbol{\varepsilon}_{k+1} \right\|^2 = \left\| \boldsymbol{\varepsilon}_k \right\|^2 - 2\eta \sum_{\Delta=1}^{N} sign(\hat{z}_{k,\Delta}) \boldsymbol{\varepsilon}_k^T (\hat{\mathbf{x}}_k \hat{d}_{k-\Delta} + \hat{\mathbf{x}}_{k-\Delta} \hat{d}_k) + \eta^2 \left\| \nabla J(\mathbf{w}_k) \right\|^2 \tag{5.14}$$

where $\hat{z}_{k,\Delta} = \hat{e}_k \hat{d}_{k-\Delta} + \hat{e}_{k-\Delta} \hat{d}_k$. Applying the expectation operator on both sides of (5.14) and letting $E\left\| \boldsymbol{\varepsilon}_{k+1} \right\|^2 < E\left\| \boldsymbol{\varepsilon}_k \right\|^2$ as in the previous case results in the following inequality.

$$\eta E\left\| \nabla J(\mathbf{w}_k) \right\|^2 < 2E \sum_{\Delta=1}^{N} \boldsymbol{\varepsilon}_k^T (\hat{\mathbf{x}}_k \hat{d}_{k-\Delta} + \hat{\mathbf{x}}_{k-\Delta} \hat{d}_k) sign(\hat{z}_{k,\Delta}) \tag{5.15}$$

Simplifying further, we get

$$\eta E\left\| \nabla J(\mathbf{w}_k) \right\|^2 < 2E \sum_{\Delta=1}^{N} \left| \hat{e}_k \hat{d}_{k-\Delta} + \hat{e}_{k-\Delta} \hat{d}_k \right| \tag{5.16}$$

Using Jensen's inequality, (5.16) can be reduced further to result in a loose upper bound on the step-size.

$$\eta E\left\|\nabla J(\mathbf{w}_k)\right\|^2 < 2\sum_{\Delta=1}^{N}\left|E[\hat{e}_k\hat{d}_{k-\Delta} + \hat{e}_{k-\Delta}\hat{d}_k]\right| \tag{5.17}$$

Notice that the RHS of (5.17) now resembles the cost function in (5.1). Rearranging the terms, we get the upper bound in (5.13).

The important point is that the bound is practical as it can be numerically computed without any knowledge of the actual filter or the noise statistics. Further, the upper bound itself can be included in the update equation to result in a normalized stochastic gradient algorithm. In general, normalization can improve speed of convergence.

Simulation Results

We will show simulation results with correlated input noise and white noise in the desired data. The framework for the simulation study is the same as we used for AEC.

System Identification with the Analytical Solution

The experimental setup is similar to the block diagram shown in Figure 5-1. We generated 50000 samples of correlated clean input signal and passed it through an unknown random FIR filter to create a clean desired signal. Gaussian random noise was passed through a random coloring filter (FIR filter with 400 taps) and then added to the clean input signal. Three different input SNR values of 5, 0 and -10dB and three different true filter lengths of 5, 10 and 15 taps were used in the experiment. For each combination of SNR value and number of taps, 100 Monte Carlo runs were performed. During each trial, a different random coloring filter as well as input/desired data was generated. We computed the Wiener solution for MSE as well as the optimal solution given by equation (5.6). The performance measure for the comparison was chosen as the error vector norm given by

$$error\ \ norm = 20\log 10\left[\left\|\mathbf{w}_T - \mathbf{w}_*\right\|\right] \tag{5.18}$$

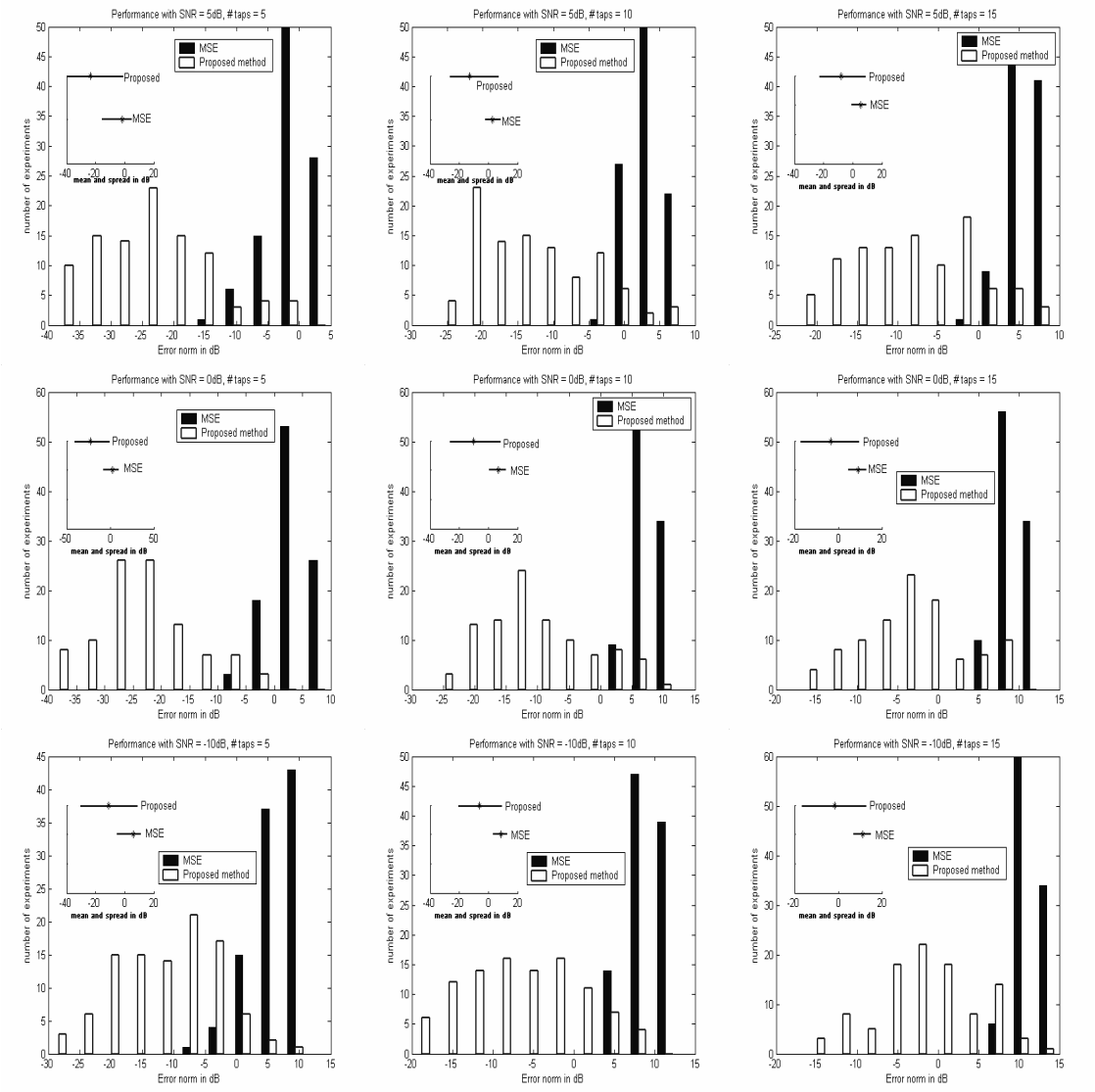


Figure 5-2. Histogram plots showing the error vector norm in dB for the proposed and MSE criteria.

where, $\mathbf{w}_*$ is the optimal solution estimated using samples and $\mathbf{w}_T$ is the true weight vector. Figure 5-2 shows the histograms of the error vector norms for the proposed method as well as MSE. The inset plots in the figure show the summary of the histograms for each method. Clearly, the performance of the new criterion is superior in every experiment given the fact that the criterion neither requires any knowledge of the noise statistics nor does it try to estimate the same from data.

System Identification with Stochastic Gradient Algorithm

We will use the stochastic gradient algorithm given by equation (5.8) to identify the parameters of a FIR filter in the presence of correlated input noise. A random four tap FIR filter was chosen as the true system. The input SNR (colored noise) was fixed at 5dB and the output SNR (white noise) was chosen to be 10dB. The step-sizes for the proposed method and the classical LMS algorithm were fixed at 1*e*-5 and 8*e*-4 respectively. One hundred Monte Carlo runs were performed and the averaged weight tracks over iterations are plotted for both algorithms in Figure 5-3. Note that our method gives a better estimate of the true parameters (shown by the square markers) than the LMS algorithm. The weight tracks of the proposed gradient method are noisier compared to those of LMS. One of the difficulties with the stochastic gradient method is the right selection of step-size. We have observed that in cases when the noise levels are very high, we require a very small step-size and hence the convergence time can be high. Additional gradient normalizations can be done to speed up the convergence. Also, the shape of the performance surface is dependent on the correlations of the input and the desired signals at different lags. If the performance surface is relatively flat around the optimal solution, we have observed that including a trivial momentum term in the update equation increases the speed of convergence.

Verification of the Local Stability of the Gradient Algorithm

In order to verify the local stability of the stochastic algorithm, we performed another experiment. This time, the four taps of the true FIR system were [0.5, -0.5, 1, -1]. The initial weights for both LMS and the gradient algorithm in (5.8) were set to the true weights. Both input and output SNR levels were kept at 10dB and the step-sizes were the same as in the previous experiment. Ideally, the algorithm in (5.8) should not move away

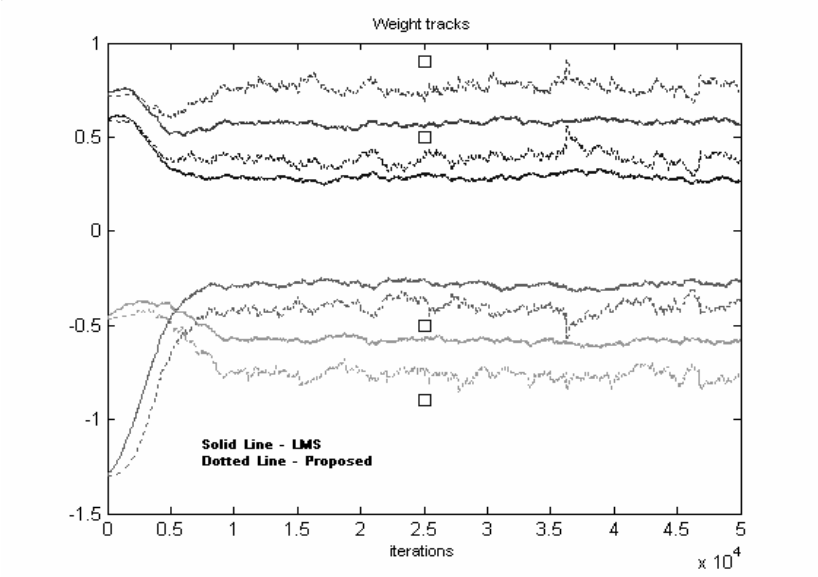


Figure 5-3. Weight tracks for LMS and the stochastic gradient algorithm in the system identification example.

from this solution as it is the global optimum. Figure 5-4 shows the weight tracks for LMS and the proposed gradient algorithm. Notice that LMS diverges from this point

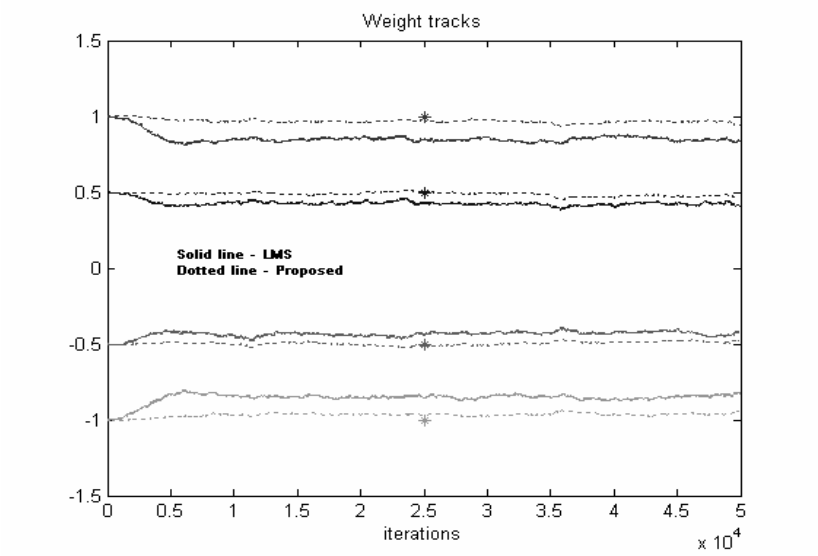


Figure 5-4. Weight tracks for LMS and the stochastic gradient algorithm showing stability around the optimal solution.

immediately and converges to a wrong (biased) solution. In comparison, the proposed algorithm shows very little displacement from the optimal solution (stable stationary

point). This proves that the algorithm is stable around the optimal solution and in effect does not diverge from this point.

So far, we have dealt with the problem of parameter estimation with correlated input noise and white noise in the desired. We will now go beyond this limitation and propose an extension to the stochastic algorithm in (5.8) to handle correlated noise in both the input and desired data.

Extensions to Correlated Noise in the Desired Data

There are a couple of approaches to solve this problem. We can assume that the noise signal in the desired data is generated by filtering a white process with an AR model (whose order is assumed to be known). This approach is similar to the IV method for colored noise. Once this assumption holds, the problem reduces to the case with correlated input noise and white noise in the desired. Then, the algorithm in (5.8) can be efficiently adopted to estimate the parameters. Notice that, in the process, we will be computing the AR coefficients that modeled the noise in the desired data as well. However, it is intuitive enough to foresee failures with this approach as the AR modeling assumption is too strong. For this reason, we will not pursue this approach further.

In order to motivate the second approach, we will first show as to why the algorithm in (5.8) cannot be used when the noise in the desired data, u_k is correlated. Adding the two terms $E[\hat{e}_k \hat{d}_{k-\Delta}]$ and $E[\hat{e}_{k-\Delta} \hat{d}_k]$ we get

$$E[\hat{e}_k \hat{d}_{k-\Delta}] + E[\hat{e}_{k-\Delta} \hat{d}_k] = \mathbf{w}_T^T \mathbf{R}_\Delta \mathbf{w}_T - \mathbf{w}_T^T \mathbf{R}_\Delta \mathbf{w} + 2E[u_k u_{k-\Delta}] \tag{5.19}$$

Observe that the last term in (5.19), which is the correlation at lag Δ is not zero for colored u_k. Therefore, the previous algorithms (both recursive and stochastic) cannot be used unless the noise correlations are known. Since the correlation structure of the noise

is not known apriori, we will include an estimate of the same in our solution. Realize that at the optimal solution $\mathbf{w}_T$, we can constrain the cost function such that the individual terms $E[\hat{e}_k \hat{d}_{k-\Delta}]$ and $E[\hat{e}_{k-\Delta} \hat{d}_k]$ converge to estimated value of $2E[u_k u_{k-\Delta}]$. In order to do so, we have to modify the cost function to include additional penalty terms. Define

$$z_\Delta(k) = e_k d_{k-\Delta} + e_{k-\Delta} d_k \tag{5.20}$$

The modified cost function is then given by

$$J(\mathbf{w}, \lambda_1 ... \lambda_\Delta ... \lambda_N, \theta_1 ... \theta_\Delta ... \theta_N) = \sum_{\Delta=1}^{N} z_\Delta^2(k) + \alpha \sum_{\Delta=1}^{N} \lambda_\Delta [z_\Delta(k) - \theta_\Delta]^2 - \alpha \sum_{\Delta=1}^{N} \lambda_\Delta^2 + \beta \sum_{\Delta=1}^{N} \theta_\Delta^2 \tag{5.21}$$

The first term $\sum_{\Delta=1}^{N} z_\Delta^2(k)$ is similar to the original cost function in (5.1) except for the squaring of the individual $z_\Delta(k)$ instead of the absolute value operator. The second term is the constraint with the Lagrangian multipliers λ_Δ defined for all lags Δ from *1* to *N*. The variable θ_Δ is an estimate of the noise correlation $2E[u_k u_{k-\Delta}]$. Ideally, this would be substituted by the noise correlation if we had apriori knowledge of the same. Notice that it is impossible to estimate the Lagrangian multipliers directly by using the constraints and the gradient of (5.21). Therefore, we have to adaptively estimate these Lagrangian multipliers too. So, the problem now becomes one of estimating a larger set of parameters $\{\mathbf{w}, \lambda_1 ... \lambda_\Delta ... \lambda_N, \theta_1 ... \theta_\Delta ... \theta_N\}$ given the input and output data. Further, we have to explicitly constrain the values of $\{\lambda_\Delta, \theta_\Delta\}_{\Delta=1,...,N}$ so that they remain bound from above. This can be achieved by including additional stabilization terms in the cost function. In (5.21), the third and fourth terms stabilize the Lagrangian multipliers and the noise correlation estimates. The constants α and β are positive real numbers that control the

stability. In vector space optimization theory, this method of imposing penalty constraints is studied under the heading *Augmented Lagrangian* techniques [16].

Now that we fully understand the structure of the cost function and the principles behind it, the next step is to the derive update rules to estimate the parameter set $\{\mathbf{w}, \lambda_1 ... \lambda_\Delta ... \lambda_N, \theta_1 ... \theta_\Delta ... \theta_N\}$. For compactness, we will refer to the parameter set as {**w**, **λ**, **θ**}, where **λ**={$\lambda_1 \ldots \lambda_N$} and similarly, **θ**={$\theta_1 \ldots \theta_N$}. We compute the following three gradients as shown.

$$\frac{\partial J(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\theta})}{\partial \mathbf{w}} = -2\sum_{\Delta=1}^{N} z_\Delta(k)(\mathbf{x}_k d_{k-\Delta} + \mathbf{x}_{k-\Delta} d_k) - 2\alpha \sum_{\Delta=1}^{N} \lambda_\Delta [z_\Delta(k) - \theta_\Delta](\mathbf{x}_k d_{k-\Delta} + \mathbf{x}_{k-\Delta} d_k) \quad (5.22)$$

$$\frac{\partial J(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\theta})}{\partial \lambda_\Delta} = \alpha [z_\Delta(k) - \theta_\Delta]^2 - 2\alpha\lambda_\Delta \quad \forall \lambda_\Delta \quad (5.23)$$

$$\frac{\partial J(\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\theta})}{\partial \theta_\Delta} = -2\alpha\lambda_\Delta [z_\Delta(k) - \theta_\Delta] + 2\beta\theta_\Delta \quad \forall \theta_\Delta \quad (5.24)$$

The optimum parameter set {**w**, **λ**, **θ**}, can be obtained by,

- Minimization with respect to **w**
- Maximization with respect to **λ**
- Minimization with respect to **θ**

The resulting update equations are given below.

$$\begin{aligned} \mathbf{w}_{k+1} &= \mathbf{w}_k - \eta_{\mathbf{w}} \frac{\partial J(\mathbf{w}_k, \boldsymbol{\lambda}_k, \boldsymbol{\theta}_k)}{\partial \mathbf{w}_k} \\ \lambda_{\Delta,k+1} &= \lambda_{\Delta,k} + \eta_\lambda \frac{\partial J(\mathbf{w}_k, \boldsymbol{\lambda}_k, \boldsymbol{\theta}_k)}{\partial \lambda_k} \\ \theta_{\Delta,k+1} &= \theta_{\Delta,k} - \eta_\theta \frac{\partial J(\mathbf{w}_k, \boldsymbol{\lambda}_k, \boldsymbol{\theta}_k)}{\partial \theta_k} \end{aligned} \quad (5.25)$$

In the above equation, the terms $\eta_w, \eta_\lambda, \eta_\theta$ are small positive step-sizes. At the optimal solution, ideally, we would have $\theta_\Delta^* = 2E[u_k u_{k-\Delta}]$, $\lambda_\Delta^* = 0$ and $\mathbf{w}^* = \mathbf{w}_T$. A rigorous proof

of convergence does not exist at this time owing to the complexity of the cost function and the interaction between the various parameters. But, we have shown through Monte Carlo simulations that the proposed algorithm approximates the actual system parameters much better than the Wiener solution for MSE or any other methods.

Experimental Results

System Identification

As we have done in the past, we will verify the working of the method in the problem of system identification. Consider a FIR system with 15 taps that needs to be estimated from noisy input (SNR = -10dB) and noisy output time series (SNR = 10dB) of length 20,000 samples. The noise terms are sufficiently colored and are assumed to be uncorrelated with each other as well as the data. One hundred Monte Carlo runs were performed and each time, the optimal Wiener MSE solution was computed for the sake of comparison. Figure 5-5 shows the error histograms computed as before for the proposed method as well as the Wiener MSE solution. Clearly, the proposed method was able to better approximate the true parameters of the FIR system.

Stochastic Algorithm Performance

We will show the performance of the stochastic gradient algorithm given by (5.25) in the presence of correlated input and output noise. Correlated noise signals were generated by randomly filtering a white process with a 1000 tap FIR filter. These were added to the clean input (sufficiently colored) and the clean desired signal that was generated by a 4-tap random FIR filter. Input and output SNR values were set to 0dB. The parameters α and β were both set to unity. The step-sizes $\eta_w, \eta_\lambda, \eta_\theta$ were 1e-4, 1e-3

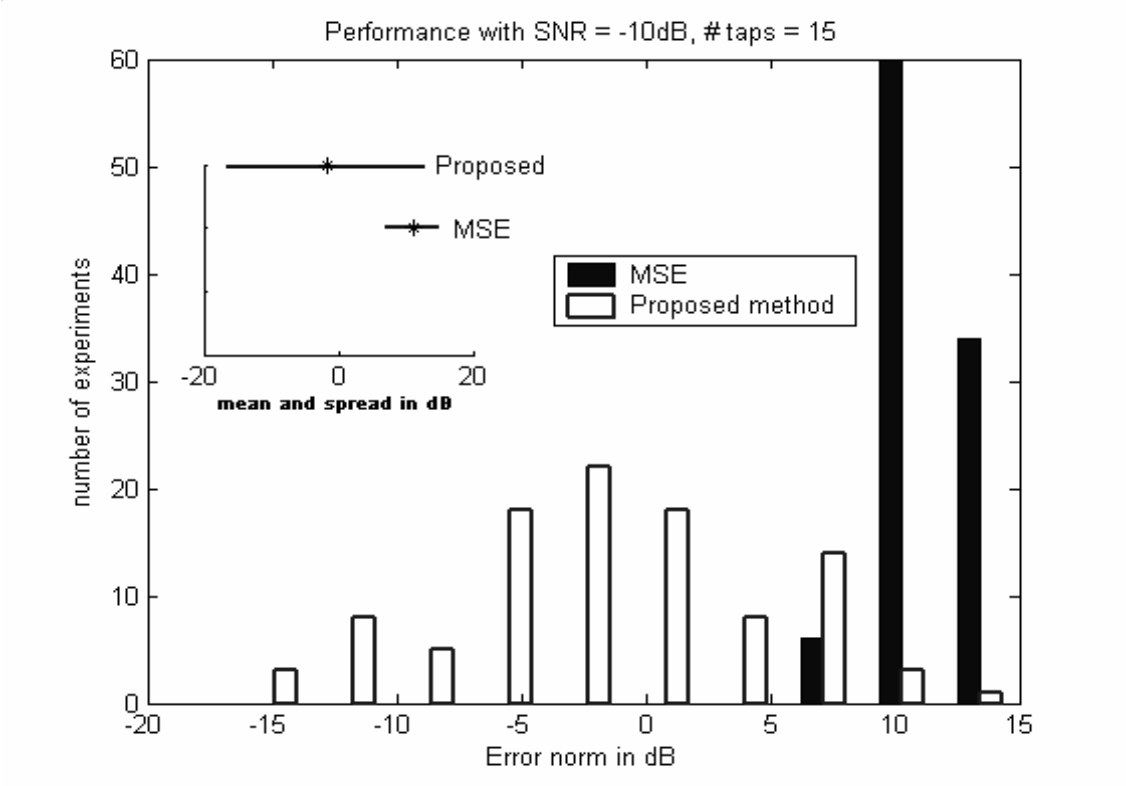


Figure 5-5. Histogram plots of the error norms for the proposed method and MSE.

and 1*e*-3 respectively. Figure 5-6 shows the averaged weight tracks over 20,000 samples. The actual FIR weights are indicated in the figure using asterisks (*). Notice that the weights estimated using the proposed stochastic algorithm converged to values very close to the true weights.

Summary

A serious limitation of the error whitening criterion discussed in the previous chapters was the assumption that the noise terms must be white. In this chapter, we presented an alternative criterion (extended the ideas of the error whitening criterion) that overcomes the limitation of the error whitening criterion. In principle, the new criterion exploits the crosscorrelation structure of the error and the desired signals in a novel way that results in the noise terms dropping out of the optimal solution. We solved the

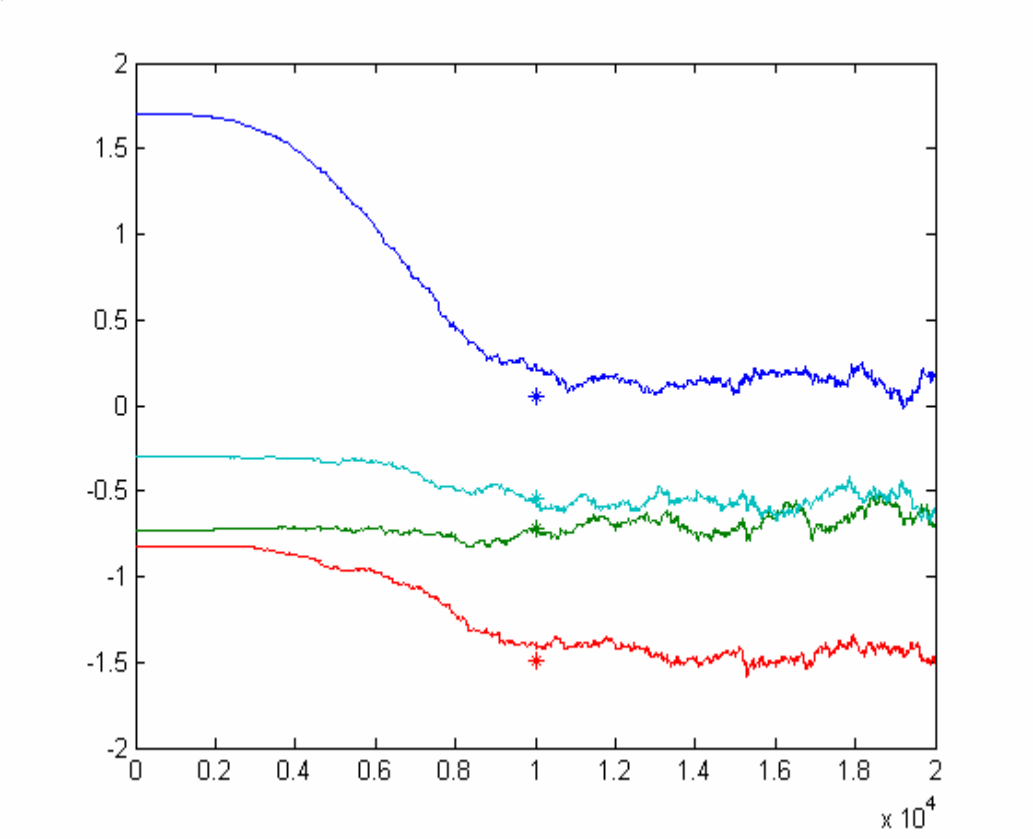


Figure 5-6. Weight tracks showing the convergence of the stochastic gradient algorithm.

problem of parameter estimation in the presence of colored noise in two steps. In the first step, we allowed only the input noise to be correlated. A new cost function was formulated, its analytical solution was derived and we also proposed a stochastic gradient algorithm. The convergence to the optimal analytical solution was mathematically established.

To extend the criterion to handle colored noise in the desired data, we introduced penalty functions in the original cost. Correlators for estimating the noise correlations noise in the desired signal) were embedded in the cost function. The principles from *Augmented Lagrangian* methods were utilized to derive stochastic gradient algorithms.

We showed simulation results in a system identification framework and verified the superiority of the proposed algorithms over other methods. In the next chapter, we will address the important issues of the performance of these new criteria when the model order of the unknown system is not known apriori.

CHAPTER 6
ON UNDERMODELING AND OVERESTIMATION ISSUES IN LINEAR SYSTEM ADAPTATION

Introduction

Till now, the objective of this dissertation has been to propose new criteria for training linear adaptive systems with noisy data. Specifically, the error whitening criterion received the major focus owing to its ability to compute unbiased parameter estimates of an unknown linear system with noisy data. In the development of the theory behind the working of this criterion, we assumed that our estimated filter is longer (has more parameters) than the actual system. In a strict sense, we always assumed a *sufficient order* for the adaptive filter. Conventionally, system identification and model-order selection have been dealt as two separate problems. In a majority of the existing work, identification has always been based on the assumption that an appropriate model order has been obtained using approaches like Akaike Information Criterion (AIC), Minimum Description Length (MDL) and many others [119,120]. However, these model-order determination methods require the data to be noise-free. In general, the problem of system identification with noisy data without apriori information about either the unknown system (model-order, linear or non-linear) or the noise statistics is ill-posed. In fact, even if we restrict the class of models to linear FIR type, there are no methods that can accurately predict the model-order with noisy data. Therefore, the *sufficient order* assumption may be costly at times depending on the application. Hence, it becomes imperative to study the performance of the criteria and the associated algorithms in

situations where the model-order is not exactly known. That is precisely our goal in this chapter: to quantify the behavior of the new criteria proposed in the previous chapters in situations where the adaptive filter has fewer parameters (undermodeling) and also in cases where it has more parameters (overestimation).

Undermodeling Effects

We will focus on the issues with undermodeling first. Let us consider an unknown linear system with N taps. Suppose we want to estimate the parameters of this system with an adaptive filter whose length is $M < N$, given noisy input and output data. The first obvious observation is that the adaptive system will never be able to approximate the true system as it does not have enough degrees of freedom. However, the interesting questions are as follows.

1. Will the reduced order adaptive filter coefficients exactly match with the first M taps of the true system? If so, under what conditions will this be happen?
2. In what sense will the reduced set of coefficients describe the true system?
3. As M approaches N, will the adaptive system response get closer to the true system response?
4. How do the answers to the above questions change if there is noise in the data?

We will now answer the above questions with relevance to the solutions obtained using both the MSE and the EWC criteria.

Consider the noise-free data case with MSE criterion for filter estimation. The classical Wiener equation gives the minimum MSE solution. It is well-known that exact coefficient matching will only occur when the input data is a white process [14]. It is very simple to mathematically prove this assertion. However, in practice, the input data is seldom white and therefore the exact coefficient matching will never occur. The coefficients of the reduced order model will try to best approximate the actual system in

the mean-squared error sense. As the length of the adaptive filter is increased, the matching between the responses of the actual and the estimated system also increases. Further, the model-mismatch between the actual system's response and the adaptive filter response decreases monotonically with increasing filter lengths. This is a very desirable property of the Wiener solution. Another interesting aspect of the Wiener solution is the principle of orthogonality [14] which states that the error signal is orthogonal to the input. Mathematically, this means

$$E[e_k \mathbf{x}_k] = \mathbf{0} \tag{6.1}$$

Assuming zero mean data, the above equation also implies that error signal and the input are uncorrelated. In the sufficient model order case, (6.1) is true for all lags of the crosscorrelation between the error and input. In the case of undermodeling, the error signal (6.1) holds only for the lags smaller than the filter length M.

Now let us consider the noisy data case. Once again, there is no exact matching of coefficients between the actual system and the estimated filter. Further, we know that the Wiener solution produces a biased estimate with noisy data. Moreover, this bias will not decrease by increasing the length of the filter. In fact, it is possible for the bias to increase as the number of coefficients increases. Although the mean-squared error monotonically decreases with increasing M, the model-mismatch does not. Thus, the nice property of the Wiener-MSE solution no longer holds in the presence of noise. Yet another downfall of Wiener solution is the fact that it changes with changing noise variance. This can lead to potentially severe situations especially in the design of inverse controllers.

We will now address the questions put forth earlier, when the parameter estimates are obtained with EWC. As in the case of Wiener solution, the optimal EWC solution

does not show exact parameter matching for real world data. Further, even in the case of undermodeling, the optimal EWC solution will tend to partially whiten the error signal. Thus the error whitening property also extends to the undermodeling scenario. However, the caveat is that, the error signal correlation is zero (or very close to zero) only at the specified lag. The uncorrelatedness of the error signal is true for all lags $L > M$ only when $M > N$. Therefore, as the length M approaches the true filter length, the higher order error correlations tend to get smaller. However, there is no guarantee that there will be a monotonic reduction in the error correlations with increasing M. Recall that in comparison to the orthogonality principle of the Wiener solution, the EWC solution satisfies a generalized orthogonality principle stated below.

$$E[e_k \mathbf{x}_{k-L} + e_{k-L} \mathbf{x}_k] = \mathbf{0} \tag{6.2}$$

Just like the Wiener solution, this property holds good only for the specified lag L in the undermodeling scenario. Further, the result becomes true for all lags when the filter order is increased beyond N.

Now consider the noisy data case. We have shown before that the optimal EWC solution will be unbiased only when the length of the estimated filter is greater than or equal to the true filter. However, EWC will still try to decorrelate the error at the specified lag. But, the error correlations at higher lags are still non-zero. When the length of the filter is increased, the values of the error correlations at the higher lags decay to zero and the model mismatch also decreases. This is a very nice property of the EWC solution that can perhaps be exploited to determine the correct model order of an unknown system. As the filter order is increased, EWC will make the error orthogonal to the input at all lags greater than the chosen lag L. This is again another property of EWC

solution which is not matched by Wiener. Perhaps the most important aspect of the EWC solution is the fact that it does not change with changes in noise variance. Thus, we conclude that most of the nice properties of Wiener MSE solution are carried over by optimal EWC solution (in a slightly different framework) even in the presence of input noise.

Overestimation Effects

The discussion on overestimation effects will be brief as most of the details are similar to the effects of undermodeling. Overestimation refers to the case when the adaptive filter length is greater than the actual filter, i.e., $M > N$. One of the major ill effects of overestimation is poor generalization even if we restrict the class of systems to simple linear FIR filters.

Consider the Wiener solution with noise-free data. The weights of the Wiener solution will contain exactly M-N zeros which is very good from the point of view of generalization. However, with noise in the input, the Wiener MSE solution will produce non-zero coefficients with noise introduced bias in every coefficient. This is because, the additional weights (in fact, the overall weight vector) tries to learn the noise in the data. Thus, generalization suffers as the estimated parameters have no physical meaning to the actual system coefficients.

In the case of EWC, we have proved that the optimal EWC solution will be unbiased in the case of overestimation. Further, we even showed that the unwanted taps are automatically zeroed out by the EWC. This is a remarkable feature of the optimal EWC solution. Further, as we said before, the solution is never dependent on the variance of the noise. We will show some simulation results highlighting some of the interesting aspects we discussed in the undermodeling and overestimation scenarios.

Experimental Results

Consider an undermodeling system identification scenario. The unknown system was a FIR filter with 4-taps (model-order 3). We used both the Wiener solution and the optimal EWC solution to estimate the parameters of this system. The input (white noise) SNR was set to 5dB and 0dB respectively. The desired signal was noise-free. Figure 6-1(left) and Figure 6-1(right) show the averaged error norms of the estimates produced by EWC and MSE criteria for 0dB and 5dB SNRs respectively. We can clearly see that both EWC and MSE solutions are biased when the number of filter taps is less than four. For four or more taps, EWC produces a much better estimate than the Wiener (the minor variations in the error norms are because of different input/output pairs used in the Monte-Carlo trials), whereas the bias in the Wiener solution does not decrease by increasing the order (unlike the noiseless case).

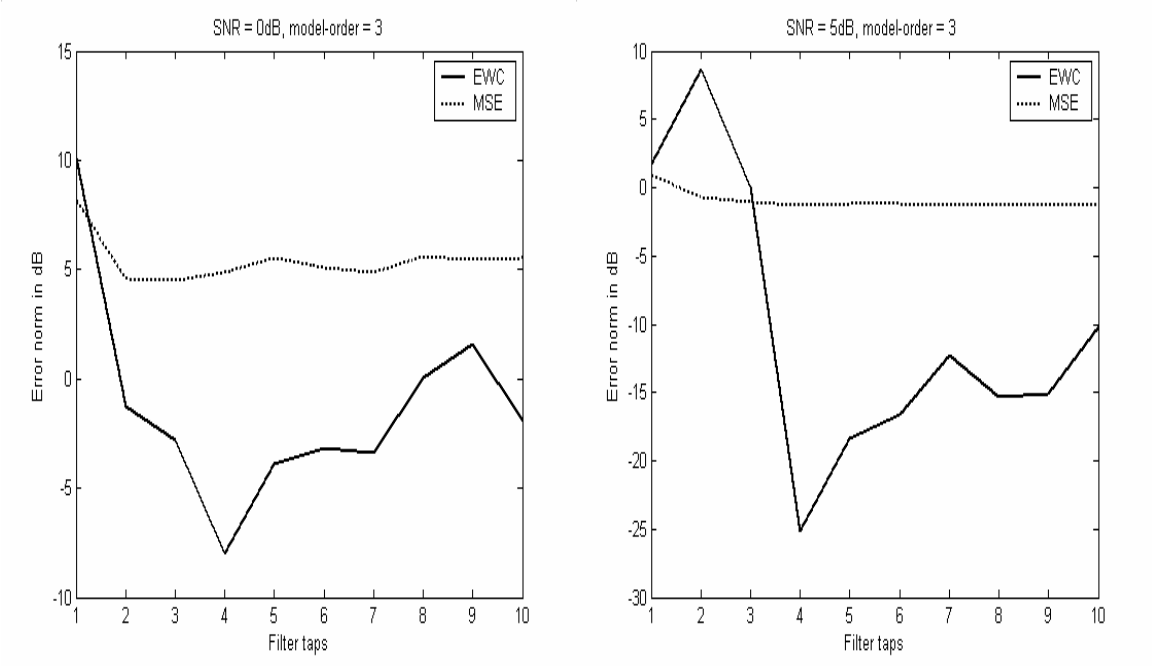


Figure 6-1. Undermodeling effects with input SNR = 0dB (left) and input SNR = 5dB (right).

We then considered another example of an unknown system with 6 taps and tried to model this system using only 2 taps. The input SNR is fixed at 0 dB. Figure 6-2 shows the plot of the crosscorrelation between input and the error. Note that the crosscorrelation is zero for only two lags in the case of MSE and with EWC, the error and the input are orthogonal only at the specified lag L=5 (arbitrarily chosen) in this example.

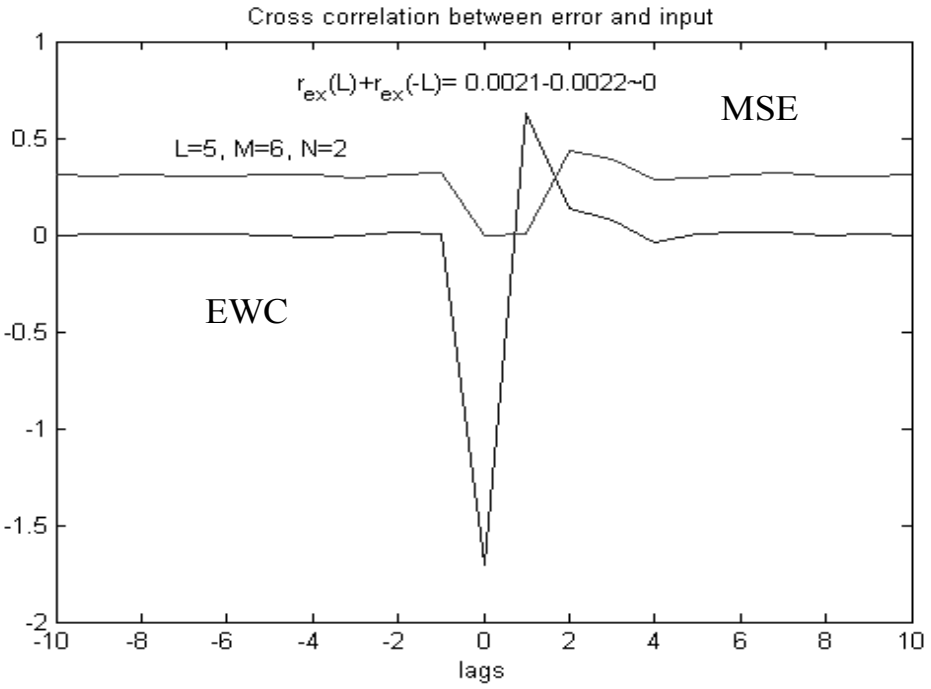


Figure 6-2. Crosscorrelation plots for EWC and MSE for undermodeling.

Figure 6-3 shows the same plot in an overestimation scenario. The key observation is that, with the MSE criterion, the error is uncorrelated with the input only for a few lags whereas in the case of EWC, error and the input are uncorrelated for all lags greater than filter length. Figure 6-4 shows the normalized error autocorrelation at higher lags in the overestimation case for both EWC and MSE. Notice that the error autocorrelations for EWC are very small for higher lags.

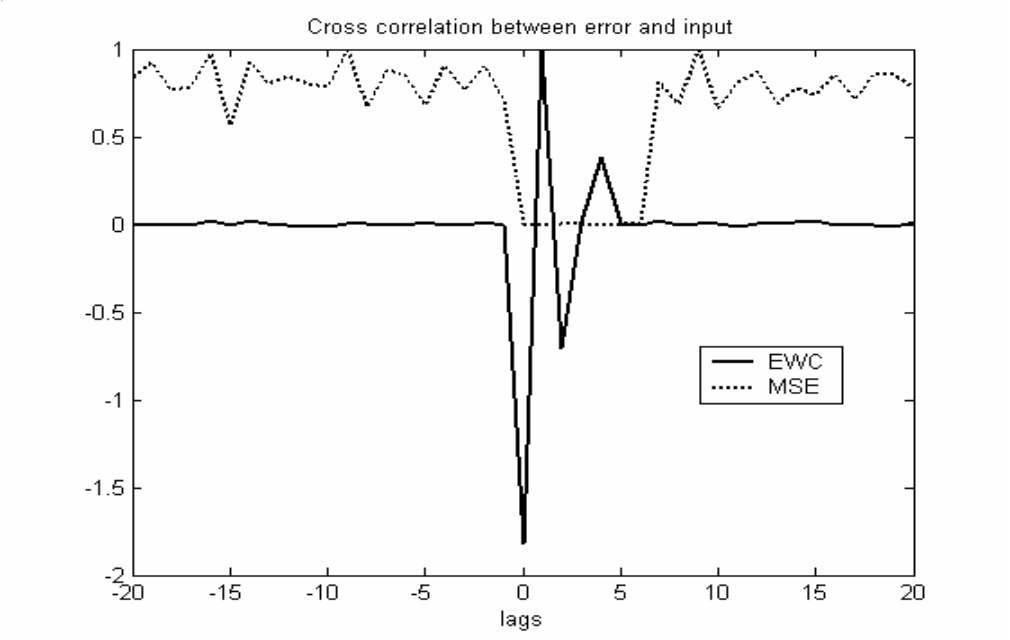


Figure 6-3. Crosscorrelation plots for EWC and MSE for overestimation.

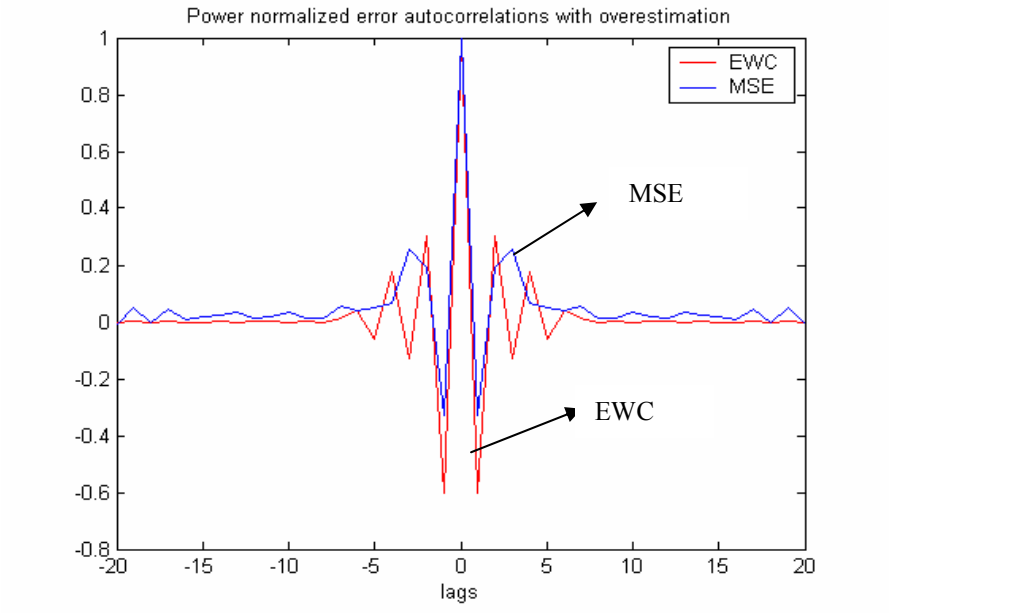


Figure 6-4. Power normalized error crosscorrelation for EWC and MSE with overestimation.

We will now verify the performance of the modified criterion discussed in Chapter 5 in the case of undermodeling. With overestimation, it is trivial to show that the additional

weights approach zero even in the presence of correlated noise. In order to understand the behavior of the proposed method in the undermodeling case, we performed a simple experiment. We chose a 4-tap FIR system and tried to model it with a 2-tap adaptive filter. Figure 6-5 shows the weight tracks for both LMS and the stochastic gradient algorithm in equation (5.8). The gradient algorithm converged to a solution that matched closely with the first two coefficients of the actual system (denoted by * in the figure). The LMS algorithm converged to an arbitrary solution that produced the minimum MSE with noisy data. This encourages us to state (speculatively) that the criterion will try to find a solution that matches the actual system in a meaningful sense. However, there is still not enough evidence to claim that the proposed method can provide exact "coefficient matching."

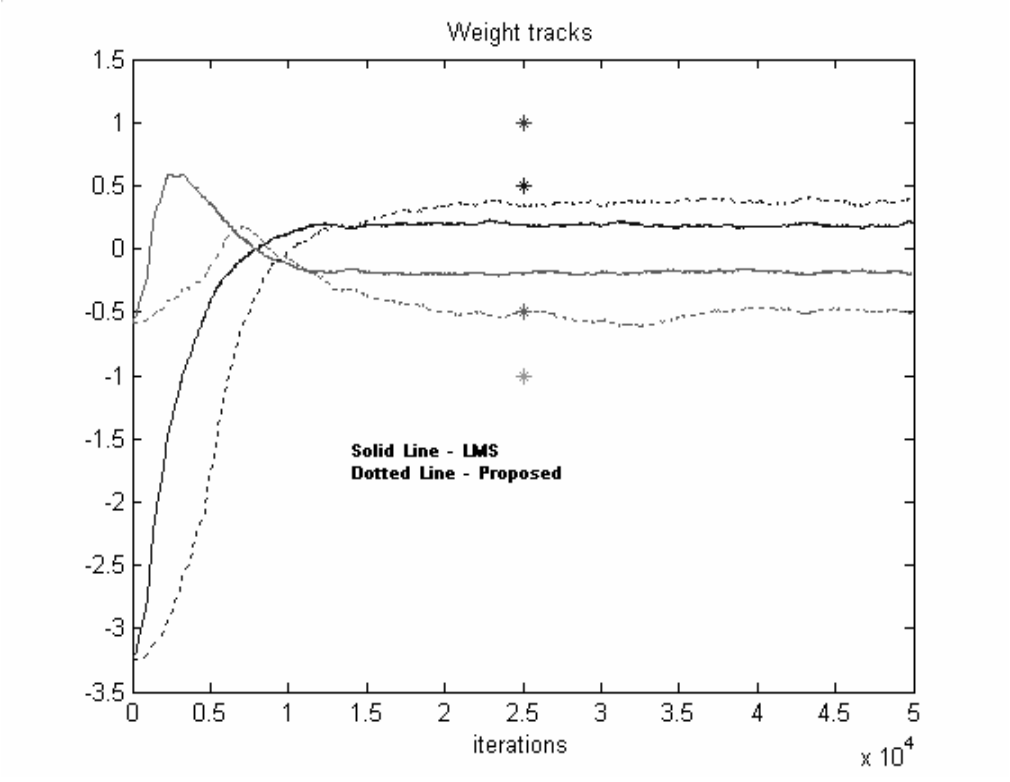


Figure 6-5. Weight tracks for LMS and the stochastic gradient algorithm in the case of undermodeling.

Summary

In this chapter, we summarized the effects of undermodeling and overestimation with EWC and MSE criteria. In essence, the error whitening criterion consistently shows good properties with and without white noise in the data, whereas the niceties of the MSE criterion are lost once noise is added to the data. One of major drawbacks of the Wiener MSE solution is its dependence on the variance of the noise whereas the same is not true for the optimal EWC solution.

We showed simulation results that quantified the observations made in this chapter. Further work is required to verify the undermodeling and overestimation performance of the modified criterion in the presence of correlated noise in the input and desired signals.

CHAPTER 7
CONCLUSIONS AND FUTURE DIRECTIONS

Conclusions

The mean-squared error criterion is by far the most widely used criterion for training adaptive systems. The existence of simple learning algorithms like LMS and RLS has promoted the applicability of this criterion to many adaptive engineering solutions. There are alternatives and enhancements to MSE that have been proposed in order to improve the robustness of learning algorithms in the presence of noisy training data. In FIR filter adaptation, noise present in the input signal is especially problematic since MSE cannot eliminate this factor. A powerful enhancement technique, total least squares, on one hand, fails to work if the noise levels in the input and output signals are not identically equal. The alternative method of subspace Wiener filtering, on the other hand, requires the noise power to be strictly smaller than the signal power to improve SNR.

We have proposed in this dissertation an extension to the traditional MSE criterion in filter adaptation, which we have named the augmented error criterion (AEC). The AEC includes MSE as a special case. Another interesting special case of the AEC is the error whitening criterion. This new criterion is inspired from the observations made on the properties of the error autocorrelation function. Specifically, we have shown that using non-zero lags of the error autocorrelation function, it is possible to obtain unbiased estimates of the model parameters even in the presence of white noise on the training data.

The AEC criterion offers a parametric family of optimal solutions. The classical Wiener solution remains a special case corresponding to the choice $\beta = 0$, whereas total noise rejection is achieved for the special choice of $\beta = -1/2$ (EWC). We have shown that the optimal solution yields an error signal uncorrelated with the predicted next value of the input vector, based on analogies with Newtonian mechanics of motion. On the other hand, the relationship with entropy through the stochastic approximation reveals a clearer understanding of the behavior of this optimal solution; the true weight vector that generated the training data marks the lags at which the error autocorrelation will become zero. We have exploited this fact to optimize the adaptive filter weights without being affected by noise.

The theoretical analysis has also been complemented by on-line algorithms that search on a sample by sample basis the optimum of the AEC. We have shown that the AEC may have a maximum, a minimum or a saddle point solution for the more interesting case of $\beta < 0$. Searching such surfaces brings difficulties for gradient descent, but search methods that use the information of the curvature work without difficulty. We have presented a recursive algorithm to find the optimum of the AEC, which is called the recursive error whitening (REW). The REW has the same structure and complexity as the RLS algorithm. We also presented gradient based algorithms to search the EWC function called EWC-LMS (and its variants) which has linear complexity $O(m)$ and requires the estimation of the sign of the update for the case $\beta = -0.5$. Theoretical conditions including step-size upper bound were derived for guaranteed convergence. Further, we showed that the gradient algorithm produces an excess error correlation that is bound from above where, the limit can be reduced by decreasing the step-size.

The optimal EWC solution is unbiased only when the input noise is white. We presented modified cost functions to handle arbitrarily correlated noise in the input and desired data. The theoretical foundations were laid and stochastic gradient algorithms were derived using *Augmented Lagrangian* methods. Convergence to the desired optimal solution was mathematically proven for a special case when only the input is allowed to have correlated noise. Finally, we briefly discussed the effects of undermodeling and overestimation with the proposed criteria.

Future Research Directions

Accurate parameter estimation with noisy data is a hard problem that has been tackled by many researchers in the past, but the resulting solutions are far from satisfactory. In this research, we proposed new criteria and algorithms to derive optimal parameter estimates. However, the methods can be effectively applied to linear feedforward systems only. Extension to nonlinear systems is not a trivial task and might require further modifications to the cost functions. It would be worthwhile to explore the advantages of using error correlation based cost functions in other engineering problems like prediction and unsupervised learning. A key part of the parameter estimation problem is the accurate determination of the model-order (linear systems only). This is a tough problem especially, with correlated noise in the data. The proposed criterion along with some sparseness constraints can be probably utilized to determine the model-order for linear systems [143,144]. For nonlinear systems, explicit regularization must be incorporated [144,145].

Instead of directly trying to derive global nonlinear models, emerging trends utilize the concept of *divide and conquer* to design multiple local linear filters that model the

nonlinear system in a piecewise manner [146-150]. The proposed criteria can be utilized to design these local models in cases when the data is noisy.

The present line of research still has open theoretical problems. Rigorous proof of convergence for the stochastic gradient algorithm outlined in equation (5.25) is yet to be provided. Further mathematical quantification of undermodeling and overestimation effects with the criteria discussed in Chapter 5 is required for a better theoretical understanding.

APPENDIX A
FAST PRINCIPAL COMPONENTS ANALYSIS (PCA) ALGORITHMS

Introduction

Principal component analysis (PCA) is a widely used statistical technique in various signal-processing applications like feature extraction, signal estimation and also detection [53-55]. There are several analytical techniques for solving the eigenvalue problem that lead to PCA [8]. These analytical techniques are block-based, computationally intensive, and are not yet appropriate for real time applications. Moreover, for many applications such as tracking where the signal statistics change over time, online solutions are more desirable. Recent research in neural networks has produced numerous iterative algorithms to solve PCA. Sanger's rule or the generalized Hebbian algorithm (GHA) [56], the Rubner-Tavan model [57,58] and the Adaptive Principal Component Extraction (APEX) model [59] which is a variation of Rubner-Tavan model are a few of them. Most of these algorithms are based on either gradient search methods or Hebbian and anti-Hebbian learning. Some lead to local implementations (APEX), which enhance their biological plausibility. The signal processing community has also been interested in iterative procedures to solve PCA. The power method, a subspace analysis technique has received a lot of attention because it estimates accurately and with fast convergence the principal eigencomponent [8,60]. Although the convergence characteristics of these methods are excellent, the update rules are non-local and computationally intensive. The PASTd [61] is another algorithm for PCA based on gradient subspace search. This algorithm is on-line and comparatively

faster than Oja's rule [62] as it uses a normalized step size. The estimation of eigenvectors and eigenvalues is therefore a well established and researched area, with many powerful results.

Brief Review of Existing Methods

Large number of existing PCA algorithms fall into one of the three categories

- Gradient based methods
- Hebbian and anti-Hebbian learning
- Subspace decompositions

Numerous cost functions are formulated and optimization techniques are applied to minimize or maximize the cost functions. The classical Oja's rule [62] is one of the first on-line rules for PCA based on Hebbian and anti-Hebbian learning. If $\mathbf{x}(n)$ denotes the input data and $y(n)$ the output after a linear transformation by the synaptic weights $\mathbf{w}$, Oja's rule for the first principal component is given by

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(y(n)\mathbf{x}(n) - y^2(n)\mathbf{w}(n)) \tag{A.1}$$

However, Oja's rule can produce only the maximum eigencomponent. Sanger [56] proposed the usage of deflation along with Oja's rule to estimate all the principal components. For a fully connected neural network, with synaptic weights w_{ji}, where, *i* is the input node and *j* is the output node, the update rule is

$$\Delta w_{ji}(n) = \eta\left[y_j(n)\mathbf{x}_i(n) - y_j(n)\sum_{k=1}^{j} w_{ki}(n)y_k(n) \right] \tag{A.2}$$

Rubner and Tavan [57,58] proposed an asymmetrical single-layer model with a lateral network among the output units. The feedforward weights are trained using the normalized Hebbian rule and lateral weights are trained using anti-Hebbian rule. This asymmetrical network performs implicit deflation. A variation of this algorithm is the

APEX algorithm proposed by Kung and Diamantaras [59]. Another widely used cost function is the reconstruction error at the output of a two-layered neural network [63], given by

$$J(\mathbf{w}) = \sum_{i=1}^{k} \beta^{k-i} \left| \mathbf{x}_i - \mathbf{w}\mathbf{w}^T \mathbf{x}_i \right|^2 \tag{A.3}$$

The scalar factor β is called the forgetting factor and $\beta < 1$. This is used to handle non-stationary data. Xu [64] has described an adaptive Principal Subspace Analysis (PSA) algorithm based on the above cost function. Xu also presents a technique to convert PSA into PCA using a symmetrical network without deflation using a scalar amplification matrix [64]. Yang [61] proposes an RLS version of the PSA technique, but uses the deflation technique instead of the scalar gain matrix. Chatterjee et.al [65] proposed a cost function similar to the one proposed by Xu, but they adopt advanced optimization techniques to solve the problem. Recently, we proposed a gradient based algorithm (and some variants) for simultaneous extraction of principal components called SIPEX [66-68]. The algorithm uses Givens rotations [8] and reduces the search space to orthonormal matrices only. Although the algorithm is fast converging, the complexity is too high. In most of the above-mentioned gradient algorithms, there is a time varying step size η involved in the update equation. The convergence and the accuracy of these algorithms heavily depend on the step-sizes, which are dependent on the eigenvalues of the data. Usually, there is an upper limit on the value of the step-size as shown in [65] and [69]. It is a non-trivial task to choose a proper step-size that is lesser than a data dependent upper bound. Subspace methods have also been used to solve PCA. Miao and Hua [70] proposed a cost function based on an information-theoretic criterion. They present a PSA algorithm, which can be used to solve the PCA problem using the standard

deflation technique. Similarly, the power method has been adopted to solve both PCA and PSA [60,71]. The power method is known to converge faster and does not involve a step-size. However, the computational burden of the power method increases with the dimensionality of the data [8].

In this appendix, we present a family of algorithms, which are as computationally tractable as the simple gradient algorithms and at the same time with the convergence rate of the subspace based algorithms. These belong to a class of fixed-point algorithms. We will first derive a new set of rules to extract the principal component and then present a rigorous convergence analysis using stochastic approximation theory. The minor components are at first estimated using the conventional deflation technique. At a later stage, we will formulate an alternative approach to estimate the minor components using robust fixed-point algorithms. Currently, the proof of convergence for the combined algorithm is under investigation. We will not provide simulation results and applications where these applications have been utilized. Effectively, the material in this appendix is a condensed version of our algorithmic contributions in the field of PCA [71-77].

Derivation of the Fixed-Point PCA Algorithm

Mathematically speaking, an eigendecomposition is the solution of the equation $\mathbf{RW} = \mathbf{W}\boldsymbol{\Lambda}$, where $\mathbf{R}$ is any real square matrix [8]. From the signal processing perspective $\mathbf{R}$ is the full covariance matrix of a zero-mean stationary random signal, $\mathbf{W}$ is the eigenvector matrix and $\boldsymbol{\Lambda}$ is the diagonal eigenvalue matrix. Without loss of generality, we will assume a zero-mean stationary signal $\mathbf{x}(n)$ with a covariance matrix $\mathbf{R} = E(\mathbf{x}_k \mathbf{x}_k^T)$. From the Rayleigh-Ritz theorem [8], the maximum eigenvalue is a stationary point of the Rayleigh quotient.

$$r(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{R} \mathbf{w}}{\mathbf{w}^T \mathbf{w}} \tag{A.4}$$

where $\mathbf{w}$ is the first principal eigenvector. Indeed, $\mathbf{w}$ is a stationary point if and only if $\frac{\partial r(\mathbf{w})}{\partial \mathbf{w}} = \mathbf{R}\mathbf{w} - \frac{\mathbf{w}^T \mathbf{R} \mathbf{w}}{\mathbf{w}^T \mathbf{w}} \mathbf{w} = \mathbf{0}$, which implies, $\mathbf{R}\mathbf{w} = \frac{\mathbf{w}^T \mathbf{R} \mathbf{w}}{\mathbf{w}^T \mathbf{w}} \mathbf{w}$. Assuming $\mathbf{w}^T \mathbf{w} = 1$, which is a property of any eigenvector, we can write, $\mathbf{R}\mathbf{w} = \left(\mathbf{w}^T \mathbf{R} \mathbf{w}\right)\mathbf{w}$, or equivalently

$$\mathbf{R}\mathbf{w} = \frac{\mathbf{w}^T \mathbf{R} \mathbf{w}}{\mathbf{w}^T \mathbf{w}} \mathbf{w} \tag{A.5}$$

Note that $\mathbf{w}^T \mathbf{R} \mathbf{w}$ is a scalar. Equation (A.5) basically states that there is a scalar relationship between $\mathbf{w}$ and its rotated version by $\mathbf{R}$. Both the numerator and the denominator can be computed as a vector matrix multiply, which is of complexity $O(N)$. Let the weight vector at iteration n, $\mathbf{w}(n)$ be the estimate of the maximum eigenvector. Then, the estimate of the new weight vector at iteration $(n+1)$ according to (A.5) is

$$\mathbf{w}(n+1) = \frac{\mathbf{R}(n)\mathbf{w}(n)}{\mathbf{w}^T(n)\mathbf{R}(n)\mathbf{w}(n)} \tag{A.6}$$

where $\mathbf{R}(n) = \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}(k)\mathbf{x}^T(k)$ is an estimate of the covariance matrix at the time step n. As a drawback, the update rule for $\mathbf{w}(n+1)$ in (A.6) tracks the eigenvalue equation assuming $\mathbf{w}^T(n+1)\mathbf{w}(n+1) = 1$ at every time step. However, note that we do not explicitly enforce this condition in (A.6). Experimental results shown in [44, 74] have proved that if we directly use (A.6) to estimate the principal component, then, we obtain convergence to a limit cycle. The norm of $\mathbf{w}(n)$ starts off initially with bounded random values, and when the weight vector approaches the eigenvector i.e., when $\mathbf{w}(n) \to \alpha \mathbf{V}$, where α is a scalar, the norm oscillates between α and $1/\alpha$. A brief

mathematical analysis of the update equation in (A.6) is presented next to get a better grasp of its behavior.

Mathematical Analysis of the Fixed-Point PCA Algorithm

In order to analyze the behavior of (A.6), we resort to the well-known stochastic approximation tools proposed by Ljung [78] and also by Kushner and Clark [79]. The idea is to associate the discrete-time adaptation rule to an ordinary differential equation (ODE). The behavior of the discrete-time algorithm is strongly or weakly tied to the stability of the ODE. Equation (A.6) is a special case of the generic stochastic approximation algorithm $\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)\mathbf{h}\big(\mathbf{w}(n), \mathbf{x}(n)\big)$. In order to apply the approximation theory, some assumptions need to be made [78-81]. We would also like to point out that (A.6) does not belong to the vanishing gain type algorithms in which the value of $\eta(n)$ is a monotonically decreasing sequence that would eventually go to zero. Equation (A.6) can be considered as a constant gain algorithm. Benveniste *et al* [80] have discussed the analysis of constant gain algorithms. Accordingly, the ODE analysis can still be applied to the constant gain algorithms with further restrictions.

5. The input is at least wide sense stationary (WSS) random process with a positive definite autocorrelation matrix **R** whose eigenvalues are distinct, positive and arranged in descending order of magnitude

6. The sequence of weight vectors $\mathbf{w}(n)$ is bounded with probability 1

7. The update function $\mathbf{h}(\mathbf{w}(n), \mathbf{x}(n))$ is continuously differentiable with respect to $\mathbf{w}(n)$ and $\mathbf{x}(n)$ and its derivatives are bounded in time

8. Even if $\mathbf{h}(\mathbf{w}(n), \mathbf{x}(n))$ has some discontinuities a mean vector field $\bar{\mathbf{h}}(\mathbf{w}, \mathbf{x}) = \lim_{n\to\infty} E\big[\mathbf{h}\big(\mathbf{w}(n), \mathbf{x}(n)\big)\big]$ exists and is regular

9. There is a locally stable solution in the Lyapunov sense to the ODE. In other words, the ODE has an attractor $\mathbf{w}^*$, whose domain of attraction is $D(\mathbf{w}^*)$

10. The weight vector $\mathbf{w}(n)$ enters a compact subset M of the basin of attraction $D(\mathbf{w}^*)$ infinitely often, with probability 1

The ODE corresponding to the update equation in (A.6) is

$$\bar{\mathbf{h}}(\mathbf{w}(t)) = \left.\frac{d\mathbf{w}(t)}{dt}\right|_{t=nT} = \left.\frac{\mathbf{w}(n+1)-\mathbf{w}(n)}{T}\right|_{T<1} = \frac{\mathbf{R}\mathbf{w}(t)}{\mathbf{w}^T(t)\mathbf{R}\mathbf{w}(t)} - \mathbf{w}(t) \tag{A.7}$$

Note that the factor T appears as a sampling interval for the forward difference approximation of continuous time derivative. This plays a crucial role in the behavior of this update equation.

Theorem 1: Consider the ODE in (A.7) and let the assumptions A.1, A.3 and A.4 hold. Then, $\mathbf{w} \to \pm\mathbf{v}_{\max}$, where $\mathbf{v}_{\max}$ is the eigenvector associated with the largest eigenvalue $\lambda_{\max}$ as $n \to \infty$ (asymptotically).

Proof: Refer to [44] for a detailed proof.

Theorem 2: The norm of the weight vector is always bounded with probability 1.

Proof: With little effort, we can see that the derivative of the norm of $\mathbf{w}(t)$ is given by

$$\frac{d\left(\mathbf{w}^T(t)\mathbf{w}(t)\right)}{dt} = \mathbf{w}^T(t)\frac{d\mathbf{w}(t)}{dt} = 1 - \mathbf{w}^T(t)\mathbf{w}(t) \tag{A.8}$$

Therefore, $d\|\mathbf{w}(t)\|^2 / dt = 2(1-\|\mathbf{w}(t)\|^2)$. We can easily solve this first order differential equation to get, $\|\mathbf{w}(t)\|^2 = 1 - \left(1-\|\mathbf{w}(0)\|^2\right)e^{-2t}$. Therefore, when $\|\mathbf{w}(0)\|^2 > 1$, $\|\mathbf{w}(t)\|^2$ is always a monotonically decreasing function and reaches unity as $t \to \infty$. If $\|\mathbf{w}(0)\|^2 < 1$, then $\|\mathbf{w}(t)\|^2$ will increase and stabilize when the norm is one. Thus, $\|\mathbf{w}(t)\|^2 \to 1$ as $t \to \infty$ as long as $\|\mathbf{w}(0)\|^2$ is bounded. It is very interesting to note that if $\|\mathbf{w}(0)\|^2 = 1$, then $\|\mathbf{w}(t)\|^2 = 1$ for all t and hence the norm becomes invariant. This might be a desirable

property for hardware implementation of the algorithm.

Theorem 3: The weight vector $\mathbf{w}(t)$ enters a compact subset *M* of the basin of attraction $D(\mathbf{w}^*)$ infinitely often, with probability 1.

Proof: From theorem 1, we know that the ODE converges to the stationary point $\mathbf{w}^* = \pm\mathbf{v}_{\max}$. Also, it is easy to show that all the other stationary points (other eigenvectors) are unstable [44]. But, this only tells us about the local behavior around the stationary points. To complete our analysis, we have to identify the domain of attraction for the stable stationary point $\mathbf{w}^*$. It is impossible in most cases to find out the domain of attraction that will span the whole weight space [82]. In order to find out the domain of attraction, we will resort to the Lyapunov function method [82]. Let $L(\mathbf{w}(t)) = 0.5\left(\mathbf{w}^T(t)\mathbf{w}(t) - 1\right)$ be a Lyapunov function defined on a region *M* consisting of all vectors such that $L(\mathbf{w}(t)) < c$, where $c > 0$. It is easy to see that $\partial L(\mathbf{w}(t))/\partial t = 1 - \|\mathbf{w}(t)\|^2$ and hence $\partial L(\mathbf{w}(t))/\partial t\big|_{\mathbf{w}=\mathbf{w}^*} = 0$. Also, if we put the constraint that $\|\mathbf{w}(0)\|^2 > 1$, then for all $t > 0$, $L(\mathbf{w}(t)) > 0$ since the minimum value of $\|\mathbf{w}(t)\|^2 = 1$. So, for all $\mathbf{w}(t) \neq \mathbf{w}^*$ in *M*, $\partial L(\mathbf{w}(t))/\partial t < 0$. Thus, the stationary point $\mathbf{w}^* = \pm\mathbf{v}_{\max}$ is globally asymptotically stable with a domain of attraction *M*. Thus, the domain of attraction *M* includes all the vectors such that $1 \le \|\mathbf{w}(t)\|^2 < 2c + 1,\ c > 0$. Obviously the stable attractor has a norm of unity and is enclosed inside *M*. Hence, as the number of iterations increases, $\mathbf{w}(t)$ will be within the set *M* and will remain inside with probability 1. From theorems 1-3, assumptions A.2, A.5 and A.6 are satisfied. We can hence deduce from the theory of stochastic approximation for adaptive algorithms with constant gains

[80] that, $\limsup_{t\to\infty} P\left\{\left\|\mathbf{w}(t)-\mathbf{v}_{\max}\right\|>\in\right\}\le C(T)$, where $\in>0$ and $C(T)$ is a small constant which becomes zero when $T\to 0$. See chapter 2 in [80] for the actual theorem statement and proof.

Earlier, we mentioned that the update equation in (A.6) enters a limit cycle when $\mathbf{w}(n)=\alpha\mathbf{v}_{\max}$ (near convergence). However, from all the above theorems, it seems like we can conclude that the update equation reaches the domain of attraction $D(\mathbf{w}^*)$ with probability 1. The contradiction arises due to the fact that we have used continuous time ODE analysis to understand the behavior of a discrete-time update equation. If the sampling interval *T* in (A.7) is not chosen to be sufficiently small, then, the ODE does not accurately represent the equation (A.6). Therefore discrepancies arise and have to be mathematically understood before being rectified.

Theorem 4: The discrete-time update equation in (A.6) enters a limit cycle when $\mathbf{w}(n)=\alpha\mathbf{v}_{\max}$, where α is any scalar constant.

Proof: As, $\mathbf{w}(n)\to\alpha\mathbf{v}_{\max}$, $\dot{\mathbf{w}}(n)=d\mathbf{w}(t)/dt\big|_{t=nT}=\mathbf{v}_{\max}\left(1-\alpha^2/\alpha\right)$, where, $\dot{\mathbf{w}}(n)$ denotes the discrete time derivative of $\mathbf{w}(n)$. Therefore, we can easily see that, the next value of the weights will be $\mathbf{w}(n+1)=\alpha\mathbf{v}_{\max}+\mathbf{v}_{\max}\left(1-\alpha^2/\alpha\right)=\mathbf{v}_{\max}/\alpha$, which is nothing but another scaled version of $\mathbf{v}_{\max}$. The derivative at this instant in time is $\dot{\mathbf{w}}(n+1)=d\mathbf{w}(t)/dt\big|_{t=(n+1)T}=-\mathbf{v}_{\max}\left(1-\alpha^2/\alpha\right)$. Notice that the derivatives at instants *n* and $(n+1)$ are the same in magnitude and opposite in sign. Therefore $\mathbf{w}(n+2)=\alpha\mathbf{v}_{\max}=\mathbf{w}(n)$. Thus, the weight vector oscillates between two values, which is clearly a case of limit cycle oscillations. To prove our point further, consider that the

non-linear function $\bar{\mathbf{h}}(\mathbf{w}(t))$ in (A.7) is smooth enough for it to be linearized in the neighborhood of the stationary point, $\mathbf{w} = \mathbf{v}_{\max}$ where $\mathbf{v}_{\max}$ is the eigenvector corresponding to the maximum eigenvalue of **R**. Thus, $\mathbf{w}(t) = \overline{\mathbf{w}} + \Delta\mathbf{w}(t)$ where, $\Delta\mathbf{w}(t)$ is a small perturbation. Then, using Taylor series expansion and retaining only the first two terms, we get $\bar{\mathbf{h}}(\mathbf{w}(t)) = \dfrac{d\mathbf{w}(t)}{dt} = \overline{\mathbf{w}} + \mathbf{A}\Delta\mathbf{w}(t)$. The matrix **A** is the *Jacobian* of the non-linear function $\bar{\mathbf{h}}(\mathbf{w}(t))$, computed at the stationary point, as $\mathbf{A} = \dfrac{\partial}{\partial \mathbf{w}(t)} \bar{h}(\mathbf{w}(t))\Big|_{\mathbf{w}=\overline{\mathbf{w}}}$. Therefore, $\dfrac{d}{dt}\Delta\mathbf{w}(t) \approx \mathbf{A}\Delta\mathbf{w}(t)$ where $\mathbf{A}$ is given by

$$\mathbf{A} = \frac{\mathbf{R}}{\mathbf{w}^T\mathbf{R}\mathbf{w}} - \mathbf{I} - \frac{2\mathbf{R}\mathbf{w}\mathbf{w}^T\mathbf{R}}{\left(\mathbf{w}^T\mathbf{R}\mathbf{w}\right)^2}\Bigg|_{\mathbf{w}=\mathbf{v}_{\max}} = \frac{\mathbf{R}}{\lambda_{\max}} - \mathbf{I} - 2\mathbf{v}_{\max}\mathbf{v}_{\max}^T \tag{A.9}$$

The nature of the equilibrium point is essentially determined by the eigenvalues of **A**.

$$\lambda_k^{\mathbf{A}} = \frac{\mathbf{q}_k^T\mathbf{R}\mathbf{q}_k}{\lambda_{\max}} - 1 - 2\mathbf{q}_k^T\mathbf{v}_{\max}\mathbf{v}_{\max}^T\mathbf{q}_k \tag{A.10}$$

Obviously, $\mathbf{q}_k$ should be an eigenvector of **R**. Hence, the eigenvalues of **A** are given by

$$\mathbf{\Lambda}^{\mathbf{A}} = diag\left[-2, \left(\frac{\lambda_2}{\lambda_{\max}} - 1\right), \left(\frac{\lambda_3}{\lambda_{\max}} - 1\right), \left(\frac{\lambda_4}{\lambda_{\max}} - 1\right) \ldots . \left(\frac{\lambda_p}{\lambda_{\max}} - 1\right)\right] \tag{A.11}$$

Note that all the eigenvalues are less than zero. So, the equilibrium point $\mathbf{w} = \mathbf{v}_{\max}$ is locally stable. However, the corresponding *z*-domain poles with $T = 1$ are given by the transformation $z - 1 = s \Rightarrow z = s + 1$.

$$z = \left[-1, \left(\frac{\lambda_2}{\lambda_{\max}}\right), \left(\frac{\lambda_3}{\lambda_{\max}}\right), \left(\frac{\lambda_4}{\lambda_{\max}}\right) \ldots . \left(\frac{\lambda_m}{\lambda_{\max}}\right)\right] \tag{A.12}$$

Clearly, a stable pole in the *s*-domain is mapped onto to the unit circle at $z = -1$. All other poles are inside the unit circle. Thus, all other modes except the mode

corresponding to the eigenvector $\mathbf{v}_{\max}$ converge asymptotically to their stable stationary points. The pole at $z = -1$ takes the discrete-time update equation in (A.6) into a limit cycle. If sampled at a higher rate, then $T < 1$ and all the *z*-poles are mapped inside the unit circle removing the limit cycling behavior. Observe that from (A.7), when the sampling interval $T < 1$, the update equation in (A.6) generalizes to

$$\mathbf{w}(n+1) = (1-T)\mathbf{w}(n) + T\frac{\mathbf{R}\mathbf{w}(n)}{\mathbf{w}^T(n)\mathbf{R}\mathbf{w}(n)} \tag{A.13}$$

It is to be noted that any value of *T* less than unity will work. Equation (A.13) is essentially a fast, fixed-point type PCA algorithm that successfully estimates the first principal component. However, the convergence speed of the algorithm is affected by the value of *T*. As the value of *T* decreases, so does the convergence speed. It will suffice to say here that the parameter T determines how well the difference equation approximates the ODE. Hence this parameter creates a trade-off between tracking and convergence with sufficient accuracy.

Self-Stabilizing Fixed-Point PCA Algorithm

The rate of convergence of (A.13) is affected by the factor T that creates an undesirable tradeoff between the speed of convergence and accuracy of the result. In this section, we will explore a variation of the algorithm given by (A.13) and present an update rule that is self-stabilizing without hurting the rate of convergence. We propose the modified update rule for the extraction of the first principal component as

$$\mathbf{w}(n+1) = \frac{\mathbf{w}(n) + \mathbf{R}(n)\mathbf{w}(n)}{1 + \mathbf{w}^T(n)\mathbf{R}(n)\mathbf{w}(n)} \tag{A.14}$$

Comparing the equations (A.13) and (A.14) we can say that both are fixed-point type algorithms that track the eigenvalue equation at every time-step. However, (A.14) does

not involve any external parameter. Typically, $\mathbf{R}$ is unknown and it has to estimated from the data. If $y(n) = \mathbf{w}^T(n)\mathbf{x}(n)$, then the rule in (A.14) can be further simplified resulting in an on-line implementation as (assuming stationarity)

$$\mathbf{w}(n+1) = \frac{\mathbf{w}(n) + \mathbf{P}(n)}{1 + Q(n)} \tag{A.15}$$

where, $\mathbf{P}(n) = \left[1 - \frac{1}{n}\right]\mathbf{P}(n-1) + \frac{1}{n}\mathbf{x}(n)y(n)$ and $Q(n) = \left[1 - \frac{1}{n}\right]Q(n-1) + \frac{1}{n}y^2(n)$. With these recursive estimators, (A.15) can be easily implemented locally. For handling non-stationary cases, a forgetting factor can be incorporated in the above recursive estimators at no additional computational cost. The overall computational complexity will still be linear in the weights i.e., *O*(*N*). The self-stabilizing feature of this algorithm can be understood by analyzing its convergence. We can adopt the same techniques that we used for the analysis of (A.6).

Mathematical Analysis of the Self-Stabilizing Fixed-Point PCA Algorithm

We will make the same set of assumptions we had before (A.1 to A.6). Again, we will let assumptions A.1, A.3 and A.4 hold without further arguments. We will now state and prove the following theorems.

Theorem 5: The only stable stationary point of the update equation in (A.14) is the principal eigenvector.

Proof: The ODE corresponding to the update equation (A.14) is given by

$$\bar{\mathbf{h}}(\mathbf{w}(t)) = \frac{d\mathbf{w}(t)}{dt} = \frac{\mathbf{w}(n+1) - \mathbf{w}(n)}{T} = \frac{\mathbf{R}\mathbf{w}(t) - \mathbf{w}(t)\mathbf{w}^T(t)\mathbf{R}\mathbf{w}(t)}{1 + \mathbf{w}^T(t)\mathbf{R}\mathbf{w}(t)} \tag{A.16}$$

$\mathbf{w}(t)$ can be expanded in terms of the complete set of orthonormal vectors (basis

vectors), as $\mathbf{w}(t) = \sum_{k=1}^{n} \theta_k(t)\mathbf{q}_k$. Substituting this in (A.16) and simplifying

$$\frac{d\theta_k(t)}{dt} = \frac{\lambda_k \theta_k(t) - \theta_k(t)\sum_{l=1}^{n} \lambda_l \theta_l^2(t)}{1 + \sum_{l=1}^{n} \lambda_l \theta_l^2(t)} \tag{A.17}$$

In (A.17), $\lambda_k, \mathbf{q}_k$ denote the k[th] eigenvalue and eigenvector of **R** respectively and $\theta_k(t)$ is the k[th] time varying projection. The dynamics of the ODE in (A.17) can be analyzed in two separate cases. In the first case, we consider $k \neq 1$. Let, $\alpha_k(t) = \theta_k(t)/\theta_1(t)$ assuming that $\theta_1(t) \neq 0$. Differentiating this wrt *t*, we get, $\frac{d\alpha_k(t)}{dt} = \frac{1}{\theta_1(t)}\frac{d\theta_k(t)}{dt} - \frac{\alpha_k(t)}{\theta_1(t)}\frac{d\theta_1(t)}{dt}$. Using (A.17)

$$\frac{d\alpha_k(t)}{dt} = -\alpha_k(t)\left[\frac{(\lambda_1 - \lambda_k)}{1 + \sum_{l=1}^{n} \lambda_l \theta_l^2(t)}\right] \tag{A.18}$$

Since, the multiplier of $\alpha_k(t)$ in (A.18) is always positive, the fixed-point of this ODE is zero for all k. In other words, $\alpha_k(t) \to 0$ as $t \to \infty$ for $k > 1$. For the case when $k = 1$, the derivative of the time varying projection is given by

$$\frac{d\theta_1(t)}{dt} = \frac{\lambda_1 \theta_1(t)[1 - \theta_1^2(t)]}{1 + \sum_{l=1}^{n} \lambda_l \theta_l^2(t)} \tag{A.19}$$

It is not easy to find the analytical solution for $\theta_1(t)$ from (A.19). However, we are interested only in the steady state solution of the ODE. To derive this, we will use a Lyapunov function $V(t)$ as $V(t) = \left[\theta_1^2(t) - 1\right]^2$. Note that $V(t) > 0$ for all *t*. Then, the derivative of $V(t)$ is simply

$$\frac{dV(t)}{dt} = -4\left[1-\theta_1^2(t)\right]\theta_1(t)\frac{d\theta_1(t)}{dt} = \frac{-4\lambda_1\theta_1^2(t)\left[1-\theta_1^2(t)\right]^2}{1+\lambda_1\theta_1^2(t)} \le 0 \qquad \text{(A.20)}$$

Hence (A.20) is stable and has a minimum given by $dV(t)/dt = 0 \Rightarrow \theta_1(t) = \pm 1$. Therefore, as $t \to \infty$, $\mathbf{w}(t) = \pm\mathbf{q}_1$, which is nothing but the principal eigenvector. When, we introduced the self-stabilizing PCA algorithm, the claim was that we would remove the external parameter *T* and still have discrete-time stability. Local stability analysis will help us get a better insight. Assuming that the non-linear function $\bar{\mathbf{h}}(\mathbf{w}(t))$ in (A.16) is smooth enough to be linearized in the neighborhood of the stable stationary point $\mathbf{w}(t) = \mathbf{q}_1$, where $\mathbf{q}_1$ is the principal eigenvector, we can compute the linearization matrix $\mathbf{A} = \left.\frac{\partial\bar{\mathbf{h}}(\mathbf{w}(t))}{\partial\mathbf{w}(t)}\right|_{\mathbf{w}=\mathbf{q}_1}$ as $\mathbf{A} = \frac{\mathbf{R} - 2\lambda_1\mathbf{q}_1\mathbf{q}_1^T - \lambda_1\mathbf{I}}{1+\lambda_1}$. The eigenvalues of $\mathbf{A}$ are given by $\lambda^{\mathbf{A}} = \left[\frac{-2\lambda_1}{1+\lambda_1}, \frac{\lambda_k - \lambda_1}{1+\lambda_1}\right], k = 2,3,4,5....,n$. Since all the poles (eigenvalues) are in the Left-Half Plane (LHP), the stationary point $\mathbf{w}(t) = \mathbf{q}_1$ is stable. The corresponding *z*-domain poles are exactly given by, $\lambda_z^{\mathbf{A}} = \left[\frac{1-\lambda_1}{1+\lambda_1}, \frac{1+\lambda_k}{1+\lambda_1}\right]$, $k = 2,3,4,5,....,n$. Note that only the first *z*-domain pole can be negative and all others are strictly positive. Also, since all the poles lie within the unit circle, the stationary point of the discrete-time update equation is also stable. In order to complete the analysis, we have to prove that the other stationary points are locally unstable. The linearization matrix $\mathbf{A}$ for the case $\mathbf{w}(t) = \mathbf{q}_k$ with $k \neq 1$ is given by $\mathbf{A} = \frac{\mathbf{R} - 2\lambda_k\mathbf{q}_k\mathbf{q}_k^T - \lambda_k\mathbf{I}}{1+\lambda_k}$. For instance, when $k = 2$, the eigenvalues of $\mathbf{A}$ are

$\lambda^{\mathbf{A}} = \left[\frac{\lambda_1 - \lambda_2}{1+\lambda_2}, \frac{-2\lambda_1}{1+\lambda_2}, \frac{\lambda_k - \lambda_2}{1+\lambda_2}\right]$, where $k = 3,4,5,6,....,n$. The first pole is in the Right-Half Plane (RHP) and hence this stationary point is locally unstable. Similarly, it can be shown that for $k > 1$, there will be exactly $k-1$ poles in the RHP that will render all these stationary points locally unstable [73]. The evolution of the discrete-time weight norm over time may not be monotonic. There is a single *z*-domain pole which can be negative if $\lambda_1 = \lambda_{\max} > 1$, and this can make the norm of the weight vector undergo damped oscillations before settling to unity (like a high pass filter). Then the upper bound on the norm is determined by the eigenvalues of the data. Further analysis is required to determine the exact upper bound.

Minor Components Extraction: Self-Stabilizing Fixed-Point PCA Algorithm

So far, we have extensively dealt with algorithms for extracting the first principal component. Although, for many applications this is sufficient, it is sometimes desirable to estimate a few minor components. Traditionally, deflation has been the key idea behind estimating the minor components. Deflation is often referred in communications literature as Graham-Schmidt Orthogonalization [83]. If we are interested in finding the second principal component, we will first subtract from the original input, the projection of the first principal component. Mathematically, if $\mathbf{x}_k$ is the actual input vector and $\mathbf{q}_1$ is the first principal component, then, after applying deflation step once, the modified input signal will be $\hat{\mathbf{x}}_k = \mathbf{x}_k - \mathbf{q}_1\mathbf{q}_1^T\mathbf{x}_k$. Now, the covariance matrix of the deflated signal is given by, $\hat{\mathbf{R}} = \mathbf{R} - \mathbf{q}_1\mathbf{q}_1^T\mathbf{R}$. Observe that, the first eigenvalue of $\hat{\mathbf{R}}$ is zero and all the other eigenvalues are the same as that of **R**. The deflation process can be sequentially applied to estimate all the minor components. By nature, deflation is a sequential

procedure, i.e., the second principal component can be estimated (converges) only after the first principal component and so on. There are a few algorithms that do not require deflation for estimating the minor components. The LMSER algorithm proposed by Xu [64] is one of them. Another algorithm is SIPEX [68] which does not require deflation as it implicitly uses an orthonormal rotation matrix. An alternative way of doing deflation is to use a lateral network as in the case of Rubner-Tavan [57,58] and APEX [54,59] algorithms. The central idea is to decorrelate the outputs of the PCA network using lateral connections between output nodes. The lateral weights are traditionally trained using inhibition learning or anti-Hebbian learning [84]. Anti-Hebbian learning is slow and can show unpredictable convergence characteristics. Choosing the step-size is tricky and usually very small step-sizes are chosen to guarantee convergence. APEX uses normalized anti-Hebbian learning, but this offers very little improvement. We propose to use the idea of lateral network; however, the learning algorithm can be derived using fixed-point theory.

In Figure A-1, we have drawn a representative 4-input, 2-output PCA network. Let $\mathbf{w}_1$ and $\mathbf{w}_2$ represent the feedforward weight vectors corresponding to the first and second output nodes respectively. The scalar weight c_1 represents the lateral connection between the first and second outputs. The correlation between the outputs is given by

$$E(y_1 y_2) = E[\mathbf{w}_1^T \mathbf{x}_k (\mathbf{w}_2^T \mathbf{x}_k + c_1 \mathbf{w}_1^T \mathbf{x}_k)] = \mathbf{w}_1^T \mathbf{R} \mathbf{w}_2 + c_1 \mathbf{w}_1^T \mathbf{R} \mathbf{w}_1 \tag{A.21}$$

If the correlation is zero, then $c_1 = -\mathbf{w}_1^T \mathbf{R} \mathbf{w}_2 / \mathbf{w}_1^T \mathbf{R} \mathbf{w}_1$, which will eventually go to zero as the weights $\mathbf{w}_1$ and $\mathbf{w}_2$ become orthogonal. Thus, the fixed-point of c_1 is zero. From this, we can deduce a fixed-point learning rule to adapt c_1 over time. This will ensure that, at every iteration, the outputs of the network are orthogonal. The learning rules for

c_1 and $\mathbf{w}_2$ are given by

$$c_1(n+1) = \frac{\mathbf{w}_1^T(n)\mathbf{R}(n)\mathbf{w}_2(n)}{1+\mathbf{w}_2^T(n)\mathbf{R}(n)\mathbf{w}_2(n)} \tag{A.22}$$

$$\mathbf{w}_2(n+1) = \frac{\mathbf{R}(n)\mathbf{w}_2(n)+\mathbf{w}_2(n)}{1+\mathbf{w}_2^T(n)\mathbf{R}(n)\mathbf{w}_2(n)} - c_1(n)\mathbf{w}_1(n) \tag{A.23}$$

Note that $c_1(n+1)$ has a different denominator term from the expression derived earlier. However, the denominator does not matter, as the fixed-point is zero. Also, from (A.23), we see that the update for $\mathbf{w}_2$ is modified by the inclusion of the $c_1(n)\mathbf{w}_1(n)$ product. In general, the update rules for both the feedforward and lateral weights are given by

$$\mathbf{w}_b(n+1) = \frac{\mathbf{R}(n)\mathbf{w}_b(n)+\mathbf{w}_b(n)}{1+\mathbf{w}_b^T(n)\mathbf{R}(n)\mathbf{w}_b(n)} - \sum_{k=1}^{b-1} c_{kb}(n)\mathbf{w}_k(n) \tag{A.24}$$

$$c_{ab}(n+1) = \frac{\mathbf{w}_a^T(n)\mathbf{R}(n)\mathbf{w}_b(n)}{1+\mathbf{w}_b^T(n)\mathbf{R}(n)\mathbf{w}_b(n)} \tag{A.25}$$

Further analysis required to quantify the gains and limitations of using the lateral network trained with these fixed-point rules.

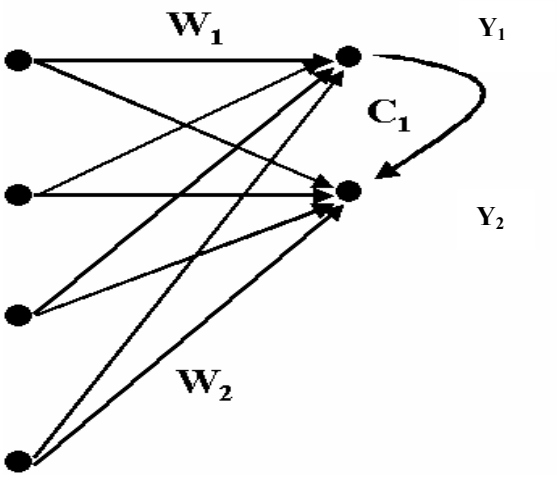


Figure A-1. Representative network architecture showing lateral connections.

APPENDIX B
FAST TOTAL LEAST-SQUARES ALGORITHM USING MINOR COMPONENTS ANALYSIS

Introduction

TLS is nothing but the solution to an over determined set of linear equations of the form $\hat{\mathbf{A}}\mathbf{x} = \hat{\mathbf{b}}$ where $\hat{\mathbf{A}}$ and $\hat{\mathbf{b}}$ denote the noisy data matrix of dimension $m \times n$ and desired vector of dimension m respectively, such that, $\left\| \Delta\hat{\mathbf{A}}; \Delta\hat{\mathbf{b}} \right\|_F = \left\| [\mathbf{A};\mathbf{b}] - [\hat{\mathbf{A}};\hat{\mathbf{b}}] \right\|_F$ is minimized or

$$(\mathbf{A};\mathbf{b})(\mathbf{x}^T;-1) = \mathbf{0} \tag{B.1}$$

Let $\mathbf{S}$ be the SVD of the augmented matrix $[\mathbf{A};\mathbf{b}]$ such that $\mathbf{S} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$, where $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4, ..., \mathbf{u}_m], \mathbf{U}^T\mathbf{U} = \mathbf{I}_m$, $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4, ..., \mathbf{v}_{n+1}], \mathbf{V}^T\mathbf{V} = \mathbf{I}_{n+1}$ and $\boldsymbol{\Sigma} = \begin{bmatrix} diag(\sigma_1, \sigma_2, \sigma_3, \sigma_4, ..., \sigma_{n+1}) \\ \mathbf{O}_{(m-n-1 \times n+1)} \end{bmatrix}$ with $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \sigma_4 \geq ...., \sigma_{n+1} > 0$. As $\sigma_{n+1} \neq 0$, in order to obtain a solution to (B.1) we must reduce the rank of $[\mathbf{A};\mathbf{b}]$ from $n+1$ to n. This can be done by making $\sigma_{n+1} = 0$ and the solution becomes

$$[\mathbf{x};-1] = -\mathbf{v}_{n+1} / \mathbf{v}_{n+1,n+1} \tag{B.2}$$

where, $\mathbf{v}_{n+1,n+1}$ is the last element of the minor eigenvector $\mathbf{v}_{n+1}$. Therefore the best approximation using (B.2) will give us $\left\| \Delta\hat{\mathbf{A}}; \Delta\hat{\mathbf{b}} \right\|_F = \sigma_{n+1}$, which means that the solution to TLS can be obtained by estimating the minimum eigenvector of the correlation matrix $\mathbf{R} = [\mathbf{A};\mathbf{b}]^T [\mathbf{A};\mathbf{b}]$ followed by the normalization as in (B.2). In the case when there is no

perturbation in **A** and **b**, then $\sigma_{n+1} = 0$ which makes $\left\|\Delta\hat{\mathbf{A}};\Delta\hat{\mathbf{b}}\right\| = 0$. When the perturbations in **A** are uncorrelated with those in **b** and when the variances of the perturbations are equal, we will still get an unbiased estimate of the parameter vector **x**. In this case the correlation matrix $\hat{\mathbf{R}} = \mathbf{R} + \varepsilon^2\mathbf{I}$, where **R** is the correlation matrix of the clean $[\mathbf{A};\mathbf{b}]$ and ε^2 is the variance of the perturbation. It is obvious that the minimum eigenvector of $\hat{\mathbf{R}}$ will be the same as the minimum eigenvector of **R** with corresponding eigenvalue equal to ε^2. Thus, the TLS solution is still unbiased. However, when the perturbation variances are not the same, then we will always have a biased estimate of the parameter vector **x**. In the next section, we will present the proposed algorithms for solving the TLS problem.

Fast TLS Algorithms

The architecture for the proposed algorithms (having complexities of $O(N)$ and $O(N^2)$ respectively) consists of a linear network with $n+1$ inputs and one or two outputs. For the $O(N)$ algorithm, we require two outputs and for the $O(N^2)$ algorithm we need only one output. We will elaborate on the details later in this section. In the input vector to the network, the first *n* elements correspond to the data input (one row of the data matrix **A**) and the last element is the corresponding desired output. The augmented input vector is represented as $\boldsymbol{\psi}(k) = [\mathbf{A}(k);\mathbf{d}(k)]^T$ where the index *k* can be time for filtering purposes. We will first describe the $O(N)$ algorithm. Let $\mathbf{W}_1, \mathbf{W}_2 \in \Re^{n+1}$ be the network weight vectors. The corresponding network outputs are $y_1(k) = \mathbf{W}_1{}^T\boldsymbol{\psi}(k),\ y_2(k) = \mathbf{W}_2^T\boldsymbol{\psi}(k)$ respectively. The goal is now to estimate the minor eigenvector of the matrix $\mathbf{R} = E(\boldsymbol{\psi}\boldsymbol{\psi}^T)$. Towards this end, we will first compute the

principal eigenvector by updating the vector $\mathbf{W}_1$ using the proposed fixed-point PCA algorithm outlined in appendix A. Accordingly, the update rule for the vector $\mathbf{W}_1$ is

$$\mathbf{W}_1(k) = \frac{\left[1-\frac{1}{k}\right]\mathbf{P}(k-1)+\left[\frac{1}{k}\right]\boldsymbol{\psi}(k)y_1(k)+\mathbf{W}_1(k-1)}{1+\left[1-\frac{1}{k}\right]Q(k-1)+\left[\frac{1}{k}\right]y_1^2(k)} \tag{B.2}$$

where, $\mathbf{P}(k) = \frac{1}{k}\sum_k \boldsymbol{\psi}(k)y_1(k)$, $Q(k) = \frac{1}{k}\sum_k y_1^2(k)$. For the TLS solution, we need the minor eigenvector. Deflation is the standard procedure to estimate the minor components. However, since we are interested in the component corresponding to the smallest eigenvalue, it would be undesirable to use deflation. We adopt a simple trick to estimate the minor eigenvector using an estimate of the maximum eigenvector [85] we obtain from (B.2). Let $\hat{\mathbf{R}} = \lambda_{\max}\mathbf{I} - \mathbf{R}$, where $\mathbf{R} = E(\boldsymbol{\psi}\boldsymbol{\psi}^T)$ as before and $\lambda_{\max}$ be the estimate of the maximum eigenvalue of $\mathbf{R}$. Note that $\hat{\mathbf{R}}$ is always positive definite and the maximum eigenvalue of $\hat{\mathbf{R}}$ is $\lambda_{\max} - \lambda_{\min}$, where $\lambda_{\min}$ is the minimum eigenvalue of $\mathbf{R}$. Hence, by estimating the maximum eigenvector of $\hat{\mathbf{R}}$, we can obtain the minimum eigenvector of $\mathbf{R}$. We can now use the fixed-point PCA rule with $\mathbf{R}$ replaced by $\hat{\mathbf{R}}$. With this modification, the update rule for $\mathbf{W}_2$ is given by

$$\mathbf{W}_2(k+1) = \left[\frac{Q(k)\mathbf{W}_2(k) - \hat{\mathbf{P}}(k) + \mathbf{W}_2(k)}{Q(k)\left\|\mathbf{W}_2(k)\right\|^2 - \hat{Q}(k)}\right] \tag{B.3}$$

where, $\hat{\mathbf{P}}(k) = \frac{1}{k}\sum_k \boldsymbol{\psi}(k)y_2(k) = \left[1-\frac{1}{k}\right]\hat{\mathbf{P}}(k-1)+\left[\frac{1}{k}\right]\boldsymbol{\psi}(k)y_2(k)$, $y_2(k) = \mathbf{W}_2{}^T(k)\boldsymbol{\psi}(k)$

and $\hat{Q}(k) = \frac{1}{k}\sum_k y_2^2(k) = \left[1-\frac{1}{k}\right]\hat{Q}(k-1)+\left[\frac{1}{k}\right]y_2^2(k)$. The definitions of $\mathbf{P}(k)$ and $Q(k)$

are still the same as before. Note that all the elements except $\|\mathbf{W}_2(k)\|$ can be computed locally. We will now prove that the algorithm in (B.3) converges to the minimum eigenvector of $\mathbf{R}$. Making the basic assumptions of stochastic approximation theory as before, we can easily write the ODE corresponding to the update equation in (B.3) as

$$\left.\frac{d\mathbf{W}_2(t)}{dt}\right|_{t=kT} = \frac{\mathbf{W}_2(t)\mathbf{W}_1^T(t)\mathbf{R}\mathbf{W}_1(t) - \mathbf{R}\mathbf{W}_2(t) + \mathbf{W}_2(t)}{1 + \mathbf{W}_1^T(t)\mathbf{R}\mathbf{W}_1(t)\|\mathbf{W}_2(t)\|^2 - \mathbf{W}_2^T(t)\mathbf{R}\mathbf{W}_2(t)} - \mathbf{W}_2(t) \tag{B.4}$$

Theorem 1: The ODE in (B.4) has a single stable stationary point $\mathbf{W}_2 = \pm\mathbf{q}_p$, where $\mathbf{q}_p$ is the eigenvector corresponding to the smallest eigenvalue of $\mathbf{R}$ with all other points locally unstable.

Proof: See [86].

We will now present the alternative $O(N^2)$ algorithm. As mentioned before, this algorithm requires a single layer linear network with one output. The input dimensionality remains the same. In order to estimate the minor eigenvector, we can utilize the fact that the maximum eigenvalue of $\mathbf{R}^{-1}$ is the same as the minimum eigenvalue of $\mathbf{R}$. Again, using the fixed-point algorithm for the principal component and utilizing matrix inversion lemma [8], we get

$$\mathbf{W}_2(k+1) = \frac{\mathbf{R}^{-1}(k+1)\mathbf{W}_2(k) + \mathbf{W}_2(k)}{\mathbf{W}_2^T(k)\mathbf{R}^{-1}(k+1)\mathbf{W}_2(k)} \tag{B.5}$$

$$\mathbf{R}^{-1}(k+1) = \mathbf{R}^{-1}(k) - \frac{\mathbf{R}^{-1}(k)\boldsymbol{\psi}(k)\boldsymbol{\psi}^T(k)\mathbf{R}^{-1}(k)}{1 + \boldsymbol{\psi}^T(k)\mathbf{R}^{-1}(k)\boldsymbol{\psi}(k)} \tag{B.6}$$

It is easy to verify that $\mathbf{W}_2(k+1)$ converges to the minimum eigenvector of $\mathbf{R}$ asymptotically with the assumption that $\mathbf{R}^{-1}(k+1) \to \mathbf{R}^{-1}$ as $k \to \infty$. The algorithm in (B.5) can be very fast when the eigenspread is very high. However, note that the

complexity of the algorithm $O(N^2)$.

We will now briefly summarize both algorithms for solving the TLS problem using the minor eigenvector. For all k with random initial conditions for $\mathbf{W}_1(0)$, $\mathbf{W}_2(0)$, build the augmented data vector $\boldsymbol{\psi}(k) = [\mathbf{A}(k); \mathbf{d}(k)]^T$.

- For $O(N)$ algorithm, compute $y_1(k) = \mathbf{W}_1^T(k)\boldsymbol{\psi}(k)$, $y_2(k) = \mathbf{W}_2^T(k)\boldsymbol{\psi}(k)$ and update $\mathbf{P}(k), Q(k), \hat{\mathbf{P}}(k), \hat{Q}(k)$
- For $O(N^2)$ algorithm, update $\mathbf{R}^{-1}$ using equation (B.6)
- For $O(N)$ algorithm update $\mathbf{W}_1(k)$ and $\mathbf{W}_2(k)$ using (B.2) and (B.3)
- For $O(N^2)$ algorithm update $\mathbf{W}_2(k)$ using (B.5)
- Compute the TLS solution given by

$$\mathbf{W}_{TLS} = -\frac{\mathbf{W}_2^*}{\mathbf{W}_{2_p}^*} \tag{B.7}$$

where, $\mathbf{W}_{2_p}^*$ denotes the last component of the vector $\mathbf{W}_2^*$. The last component of $\mathbf{W}_{TLS}$ will be –1 and is discarded. Thus $\mathbf{W}_{TLS}$ will be of dimension n and not $n+1$.

Simulation Results with TLS

We will now show the performance of the proposed TLS algorithms through simulations.

Simulation 1: Noise Free FIR Filter Modeling

We generated a 200-tap FIR filter with random filter coefficients. The input was a 1000 sample length random signal. Both the input and the desired response (which is just the filtered input) were noise free. Thus, the minimum eigenvalue of the composite signal $\boldsymbol{\psi}(k)$ must be zero. Figure B-1 shows the plots of the estimation of the minimum eigenvalue using the $O(N^2)$ algorithm along with the direction cosine defined as

$$DC(k) = \frac{\mathbf{W}_2^T(k)\mathbf{V}_p}{\left\|\mathbf{W}_2(k)\right\|\left\|\mathbf{V}_p\right\|} \tag{B.8}$$

where, $\mathbf{V}_p$ is the true minor eigenvector. Note that the algorithm converges in just 300 on-line iterations. This high convergence speed can be attributed to the fact that the algorithm performs better when the eigenspread is very high (infinity, in this case) However, the computational load becomes very significant with 201 dimensions in the composite signal. For comparisons with other methods see [86].

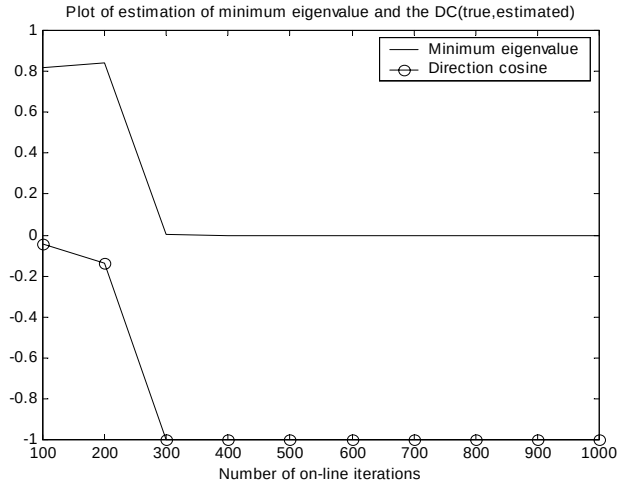


Figure B-1. Estimation of minor eigenvector.

Simulation 2: FIR Filter Modeling with Noise

As stated before, the advantage of TLS lies in the fact that it can provide unbiased estimates when the observations are noisy. In this simulation, we will experimentally verify this fact and show that RLS can at best provide a biased estimate of the parameters. We will consider a FIR with 50-taps and use the $O(N)$ rule to estimate the minimum eigenvector. The input is a colored noise signal with 15000 samples. The desired signal is the filtered input signal. Uncorrelated random noise with unit variance is added to the input and desired signals. Note that in this case, the minimum eigenvalue of

the composite signal will be equal to the variance of the noise, i.e., 1. Figure B-2 shows the convergence of the minimum eigenvalue. The algorithm converges in less than 15,000 iterations, which is one complete presentation of the input data. Figure B-3 shows the comparison between the estimated filter coefficients and the true filter coefficients using the $O(N)$ rule and Figure B-4 depicts the performance of RLS. It is clearly seen that in the presence of noise, RLS estimates are always biased and this bias can reach unacceptable levels when the noise variance increases.

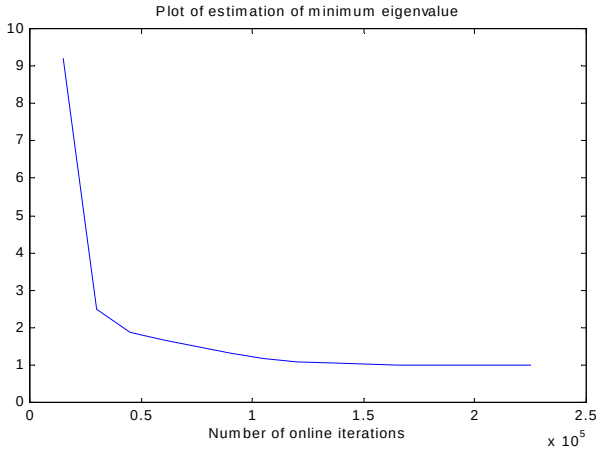


Figure B-2. Minimum eigenvalue estimation.

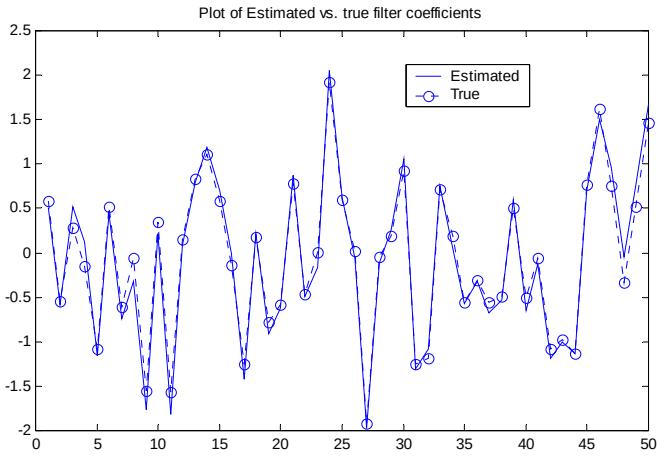


Figure B-3. Comparison between the estimated and true filter coefficients using TLS.

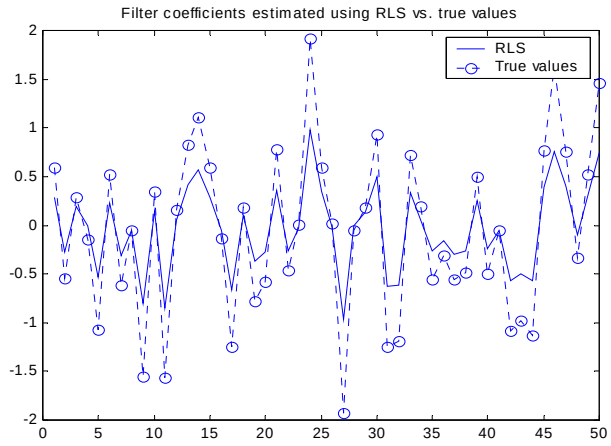


Figure B-4. Comparison between the estimated and true filter coefficients using RLS.

We can extend the application of TLS technique to model IIR filters also. In this case, we can use the past values of the desired responses along with the input to form the composite signal. The architecture will remain the same as that of the FIR case.

APPENDIX C
ALGORITHMS FOR GENERALIZED EIGENDECOMPOSITION

Introduction

In appendix A, we discussed about Principal Components Analysis (PCA) as optimal adaptive matrix transformations. PCA by nature, involves estimation of optimal transformations (also referred to as projections and rotations) derived to span the space of the input data. However, there are several applications, where we are forced to find projections in joint spaces. One simple application is the classical pattern classification problem with two classes where the goal is to determine the best discriminant in the joint space that separates the two classes. This is well known in the pattern recognition literature as the *Fisher* discriminant [53,94], which brings us to the concept of generalized eigendecomposition (GED). Formally speaking, GED solves the generalized eigenvalue equation of the matrix pencil (**A**,**B**), which is given by $\mathbf{AV} = \mathbf{BV\Lambda}$ [8]. Note that, when **B** is identity matrix, then, the generalized eigenvalue problem boils down to the PCA problem. In the Fisher discriminant case, the matrices **A** and **B** are the *between-class* scatter and the *within-class* scatter respectively. Like PCA, GED is an extremely useful statistical tool and has many applications including feature extraction, pattern classification, signal estimation and detection [53,55].

Review of Existing Learning Algorithms

Many analytical techniques have been developed in the linear algebra literatures to compute the generalized eigenvectors [8]. These numerical techniques are computationally prohibitive and moreover they require blocks of data. For engineering

applications, on-line, sample-by-sample algorithms are desired. The importance of the sample-by-sample methods is even more pronounced in environments where signal statistics change slowly over time and hence tracking becomes a key issue. Only fast, on-line algorithms can adapt quickly to the changing environment while block techniques lack this feature. Compared to PCA, there are fewer on-line algorithms for GED. Mao and Jain have proposed a two-step PCA approach to solve GED in [55]. They use the Rubner-Tavan model [57,58] for training the PCA blocks. Thus the convergence of their method depends solely on the convergence of the PCA algorithms. A similar approach is used in [71,74] but a faster PCA algorithm has been incorporated that drastically improves the performance. However, as mentioned before, this two-step PCA method may not be very suitable for real-world applications. Chatterjee *et al* have proposed a gradient algorithm based on linear discriminant analysis (LDA) [69]. They propose an on-line algorithm for extracting the first generalized eigenvector and then use deflation procedure for estimating the minor components. They prove convergence of the algorithm using stochastic approximation theory. However, the main drawback of their method is that the algorithm is based on simple gradient techniques and this makes convergence dependent on the step-sizes that are difficult to set apriori. Xu *et al* have developed an on-line and local algorithm for GED [95]. The rule for extracting the first generalized eigenvector is similar to the LDA algorithm in [69], but they use a lateral inhibition network similar to the APEX algorithm for PCA [59] for extracting the minor components. Although the problem formulation is novel, there is no rigorous proof of global convergence. A quasi-Newton type algorithm was proposed by Mathew *et al* [96]. The computational complexity is quite high, but a pipelined architecture can be used to

reduce the complexity [96]. The method makes approximations in computing the Hessian required for the Newton-type methods. Diamantaras *et al* demonstrate an unsupervised neural model based on the APEX models for extracting first generalized eigenvector only [97]. The update equations are quite complicated given the fact that it extracts only the principal generalized eigenvector. Most of the above mentioned algorithms are based on gradient methods and they involve the selection of right step-sizes to ensure convergence. In general the step-sizes have an upper bound that is a function of the eigenvalues of the input data. This fact makes it very hard on many occasions to choose the proper step-size. On the other hand, we can adopt better optimization procedures, but computational complexity is also a key issue. Motivated by the success of the fixed-point PCA algorithm discussed in appendix A, we will derive a fixed-point GED algorithm based on the same lines.

Fixed-Point Learning Algorithm for GED

From a mathematical perspective, generalized eigendecomposition involves solving the matrix equation $\mathbf{R_1 W = R_2 W \Lambda}$, where $\mathbf{R_1, R_2}$ are square matrices, $\mathbf{W}$ is the generalized eigenvector matrix and $\mathbf{\Lambda}$ is the diagonal generalized eigenvalue matrix [8]. These are typically the full covariance matrices of zero-mean stationary random signals $x_1(n)$ and $x_2(n)$ respectively. For real symmetric and positive definite matrices, all the generalized eigenvectors are real and the corresponding generalized eigenvalues are positive. GED possesses some very interesting properties that can be exploited for various signal-processing applications. The generalized eigenvectors achieve simultaneous diagonalization of the matrices $\mathbf{R_1, R_2}$ as $\mathbf{W}^T \mathbf{R}_1 \mathbf{W} = \mathbf{\Lambda}$ and $\mathbf{W}^T \mathbf{R}_2 \mathbf{W} = \mathbf{I}$. This property enables us to derive an iterative algorithm for GED using

two PCA steps as mentioned in the previous section. Alternatively, GED is also referred to as Oriented PCA (OPCA) [59,97]. Accordingly the generalized eigenvectors act as filters in the joint-space of the two signals $x_1(n)$ and $x_2(n)$, minimizing the energy of one of the signals and maximizing the energy of the other at the same time. This property has been successfully applied to the problems of signal separation [98] and more recently for detecting transitions in time series [74]. The oriented energy concept comes from the fact that the generalized eigenvalues can be expressed as ratios of two energies. Equivalently, this means that any generalized eigenvector $\mathbf{w}$ that is a column of the matrix $\mathbf{W}$ is a stationary point of the function

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{R}_1 \mathbf{w}}{\mathbf{w}^T \mathbf{R}_2 \mathbf{w}} \tag{C.1}$$

This is because $\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \frac{\mathbf{w}^T \mathbf{R}_2 \mathbf{w}(2\mathbf{R}_1 \mathbf{w}) - \mathbf{w}^T \mathbf{R}_1 \mathbf{w}(2\mathbf{R}_2 \mathbf{w})}{\left(\mathbf{w}^T \mathbf{R}_2 \mathbf{w}\right)^2} = 0 \Rightarrow \mathbf{R}_1 \mathbf{w} = \frac{\mathbf{w}^T \mathbf{R}_1 \mathbf{w}}{\mathbf{w}^T \mathbf{R}_2 \mathbf{w}} \mathbf{R}_2 \mathbf{w}$.

This is nothing but the generalized eigenvalue equation and the generalized eigenvalues are the values of (C.1) evaluated at the stationary points. Most of the gradient-based methods use equation (C.1) as the cost function and perform maximization with some constraints. It is easy to recognize the fact that the well known linear discriminant analysis or LDA problem involves maximizing the ratio in (C.1) with $\mathbf{R}_1 = \mathbf{\Sigma}_B$, the between-class covariance matrix and $\mathbf{R}_2 = \mathbf{\Sigma}_w$ which is the within-class covariance matrix. We will now state our approach to estimate the generalized eigenvector corresponding to the largest generalized eigenvalue hereafter referred to as the principal generalized eigenvector. Using (C.1), we can rewrite the GED equation as

$$\mathbf{R}_1 \mathbf{w} = \frac{\mathbf{w}^T \mathbf{R}_1 \mathbf{w}}{\mathbf{w}^T \mathbf{R}_2 \mathbf{w}} \mathbf{R}_2 \mathbf{w} \tag{C.2}$$

If $\mathbf{R}_2 = \mathbf{I}$, then (C.2) reduces to the Rayleigh quotient and the generalized eigenvalue problem will degenerate to PCA. Left multiplying (C.2) by $\mathbf{R}_2^{-1}$ and rearranging the terms, we get

$$\mathbf{w} = \frac{\mathbf{w}^T \mathbf{R}_2 \mathbf{w}}{\mathbf{w}^T \mathbf{R}_1 \mathbf{w}} \mathbf{R}_2^{-1} \mathbf{R}_1 \mathbf{w} \tag{C.3}$$

Equation (C.3) is the basis of our iterative algorithm. Let the weight vector $\mathbf{w}(n-1)$ at iteration $(n-1)$ be the estimate of the principal generalized eigenvector. Then, the estimate of the new weight vector at iteration n according to (C.3) is

$$\mathbf{w}(n) = \frac{\mathbf{w}^T(n-1)\mathbf{R}_2(n)\mathbf{w}(n-1)}{\mathbf{w}^T(n-1)\mathbf{R}_1(n)\mathbf{w}(n-1)} \mathbf{R}_2^{-1}(n)\mathbf{R}_1(n)\mathbf{w}(n-1) \tag{C.4}$$

We can observe that (C.4) tracks the GED equation at every time-step. The fixed-point algorithms are known to be faster compared to the gradient algorithms, but many fixed-point algorithms work in batch-mode, which means that the weight update is done after a window of time [99]. This can be a potential drawback of the fixed-point methods, but in our case, we can easily transform the fixed-point update in (C.4) into a form that can be implemented online. To begin with, we need a matrix inversion operation for each update. By using Sherman-Morrison-Woodbury matrix inversion lemma [8] we get

$$\mathbf{R}_2^{-1}(n) = \mathbf{R}_2^{-1}(n-1) - \frac{\mathbf{R}_2^{-1}(n-1)\mathbf{x}_2(n)\mathbf{x}_2^T(n)\mathbf{R}_2^{-1}(n-1)}{1 + \mathbf{x}_2^T(n)\mathbf{R}_2^{-1}(n-1)\mathbf{x}_2(n)} \tag{C.5}$$

If we assume that $\mathbf{w}$ is the weight vector of a single-layer feed-forward network, then define $y_1(n) = \mathbf{w}^T(n-1)\mathbf{x}_1(n)$ and $y_2(n) = \mathbf{w}^T(n-1)\mathbf{x}_2(n)$ as the outputs of the network for signals $\mathbf{x}_1(n)$ and $\mathbf{x}_2(n)$ respectively. With this definition, it is easy to show that, $\mathbf{w}^T(n-1)\mathbf{R}_1(n)\mathbf{w}(n-1) = \frac{1}{n}\sum_{i=1}^{n} y_1^2(i)$, $\mathbf{w}^T(n-1)\mathbf{R}_2(n)\mathbf{w}(n-1) = \frac{1}{n}\sum_{i=1}^{n} y_2^2(i)$. This is true

in the stationary cases when sample-variance estimators can be used instead of the expectation operators (of course assuming that the weights are changing slowly enough). However, for non-stationary signals, a simple forgetting factor can be included with a trivial change in the update equation. With these simplifications, we can write the modified update equation for the stationary case as

$$\mathbf{w}(n) = \frac{\sum_{i=1}^{n} y_2^2(i)}{\sum_{i=1}^{n} y_1^2(i)} \mathbf{R}_2^{-1}(n) \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_1(i) y_1(i), \quad y_l(i) = \mathbf{w}^T(i-1)\mathbf{x}_l(i), \quad l = 1, 2 \qquad \text{(C.6)}$$

where $\mathbf{R}_2^{-1}(n)$ is estimated using (C.5). In order to implement the summations, we can use recursive estimators. We will now summarize the fixed-point algorithm below.

- Initialize the $\mathbf{w}(0) \in \Re^{n\times 1}$ to a random vector
- Initialize a vector $\mathbf{P}(0) \in \Re^{n\times 1}$ to a vector with small random values
- Fill the matrix $\mathbf{Q}(0) \in \Re^{n\times n}$ with small random values
- Initialize scalar variables $C_1(0), C_2(0)$ to zero
- For $j > 0$
- Compute $y_1(j) = \mathbf{w}^T(j-1)\mathbf{x}_1(j)$ and $y_2(j) = \mathbf{w}^T(j-1)\mathbf{x}_2(j)$
- Update **P** as $\mathbf{P}(j) = \left[1 - \frac{1}{j}\right]\mathbf{P}(j-1) + \left[\frac{1}{j}\right]\mathbf{x}_1(j)y_1(j)$
- Update **Q** as $\mathbf{Q}(j) = \mathbf{Q}(j-1) - \frac{\mathbf{Q}(j-1)\mathbf{x}_2(j)\mathbf{x}_2^T(j)\mathbf{Q}(j-1)}{1 + \mathbf{x}_2^T(j)\mathbf{Q}(j-1)\mathbf{x}_2(j)}$
- Update C_1, C_2 as $C_i(j) = \left[1 - \frac{1}{j}\right]C_i(j-1) + \left[\frac{1}{j}\right]y_i^2(j), \; i = 1,2$
- Update the weight vector as $\mathbf{w}(j) = \frac{C_2(j)}{C_1(j)}\mathbf{Q}(j)\mathbf{P}(j)$
- Normalize the weight vector
- Go back to step 5 and repeat until convergence is reached

The above algorithm extracts the principal generalized eigenvector. For the minor components, we will resort to the deflation technique. Consider the following pair of

matrices, $\hat{\mathbf{R}}_1 = \left[\mathbf{I} - \frac{\mathbf{R}_1\mathbf{w}_1\mathbf{w}_1^T}{\mathbf{w}_1^T\mathbf{R}_1\mathbf{w}_1}\right]\mathbf{R}_1, \hat{\mathbf{R}}_2 = \mathbf{R}_2$, where $\mathbf{w}_1$ is the best estimate of the principal generalized eigenvector using (C.6). For this pair of matrices, $\hat{\mathbf{R}}_1\mathbf{w}_1 = 0$ and $\hat{\mathbf{R}}_1\mathbf{w}_i = \lambda_i\mathbf{R}_2\mathbf{w}_i,\ i > 1$. The time index n is implicit and is omitted for convenience. It is easy to see that, $\hat{\mathbf{R}}_1 = \mathbf{R}_1 - \frac{\mathbf{R}_1\mathbf{w}_1\mathbf{w}_1^T\mathbf{R}_1}{\mathbf{w}_1^T\mathbf{R}_1\mathbf{w}_1} = \mathbf{R}_1 - 2\frac{\mathbf{R}_1\mathbf{w}_1}{\mathbf{w}_1^T\mathbf{R}_1\mathbf{w}_1}\mathbf{w}_1^T\mathbf{R}_1 + \frac{\mathbf{R}_1\mathbf{w}_1}{\mathbf{w}_1^T\mathbf{R}_1\mathbf{w}_1}\mathbf{w}_1^T\mathbf{R}_1$. With, $y_1 = \mathbf{w}_1^T\mathbf{x}_1(n)$, $\hat{\mathbf{R}} = E\left\{\left(\mathbf{x}_1 - \frac{\mathbf{R}_1\mathbf{w}_1}{\mathbf{w}_1^T\mathbf{R}_1\mathbf{w}_1}y_1\right)\left(\mathbf{x}_1^T - \frac{\mathbf{w}_1^T\mathbf{R}_1}{\mathbf{w}_1^T\mathbf{R}_1\mathbf{w}_1}y_1\right)\right\} = E\{\hat{\mathbf{x}}_1\hat{\mathbf{x}}_1^T\}$, where, $\hat{\mathbf{x}}_1(n)$ is given by

$$\hat{\mathbf{x}}_1(n) = \mathbf{x}_1(n) - \frac{\mathbf{R}_1\mathbf{w}_1}{\mathbf{w}_1^T\mathbf{R}_1\mathbf{w}_1}y_1(n) = \mathbf{x}_1(n) - \frac{y_1(n)\sum_{i=1}^{n}\mathbf{x}_1(i)y_1(i)}{\sum_{i=1}^{n}y_1^2(i)} \tag{C.7}$$

With the above deflation procedure, we can estimate the second generalized eigenvector using the same update rule in (C.6) with $\mathbf{x}_1(n)$ replaced by $\hat{\mathbf{x}}_1(n)$. $\mathbf{x}_2(n)$ remains the same as before. At this point, we would like to stress that the deflation scheme does not require any further computations as the summations $\sum_{i=1}^{n}\mathbf{x}_1(i)y_1(i)$ and $\sum_{i=1}^{n}y_1^2(i)$ are already pre-calculated for estimating the principal generalized eigenvector. They can be efficiently substituted by $\mathbf{P}(j)$ and $C_1(j)$ mentioned in the summary of the algorithm for the principal generalized eigenvector. The fixed-point GED algorithm has many useful properties compared to other methods. The convergence of the algorithm is exponential whereas the convergence of the on-line gradient methods is linear. This is generally true for most fixed-point algorithms [99]. Simulations have shown beyond doubt that the proposed algorithm has superior convergence speed when compared with other methods.

As we have seen before, gradient algorithms are dependent on step-sizes, which results in non-robust performance. In contrast, the fixed-point algorithm does not require a step-size for the updates (similar to the PCA algorithm in appendix A). Moreover, like the gradient methods, the fixed-point algorithm has an on-line implementation that is computationally feasible. The computational complexity is $O(N^2)$ where N is the dimensionality of the data, which is comparable to the complexities of the algorithms in [55,69].

Mathematical Analysis

We will now investigate the convergence characteristics of the GED algorithm given by (C.6) using stochastic approximation techniques that was cited before in the analysis of PCA algorithms. Without dwelling too much into the methodology of stochastic approximation tools, we directly apply it to our algorithm. We will state some assumptions similar to the ones used in PCA algorithm analysis.

- The inputs $\mathbf{x}_1(n), \mathbf{x}_2(n)$ are at least wide sense stationary (WSS) with positive definite autocorrelation matrices $\mathbf{R}_1, \mathbf{R}_2$.
- The sequence of weight vectors $\mathbf{w}(n)$ is bounded with probability 1.
- The update function $\mathbf{h}(\mathbf{w}(n), \mathbf{x}_1(n), \mathbf{x}_2(n))$ is continuously differentiable with respect to $\mathbf{w}$, $\mathbf{x}_1, \mathbf{x}_2$ and its derivatives are bounded in time.
- Even if $\mathbf{h}(\mathbf{w}(n), \mathbf{x}_1(n), \mathbf{x}_2(n))$ has some discontinuities a mean vector field $\bar{\mathbf{h}}(\mathbf{w}, \mathbf{x}_1, \mathbf{x}_2) = \lim_{n\to\infty} E[\mathbf{h}(\mathbf{w}(n), \mathbf{x}_1(n), \mathbf{x}_2(n))]$ exists and is regular.
- The initial weights are chosen such that $\mathbf{w}^T(0)\mathbf{q}_1 \neq 0$, where $\mathbf{q}_1$ is the principal generalized eigenvector.

The second assumption is satisfied by the fact that we force the norm of the weight vector to unity. Thus, when the weight vector is bounded, then the updates are also bounded in

time. Since we assume that the matrices $\mathbf{R}_1, \mathbf{R}_2$ are full rank, the inverses exist. By satisfying the first two assumptions, it is easy to see that the derivatives of the update function are also bound in time. Under these conditions, we enunciate the following theorem.

Theorem 1: There is a locally stable solution in the Lyapunov sense to the ODE. In other words, the ODE has an attractor $\mathbf{w}^*$, whose domain of attraction is $D(\mathbf{w}^*)$.

Proof: The update function $\bar{\mathbf{h}}(\mathbf{w}, \mathbf{x}_1, \mathbf{x}_2) = \lim_{n\to\infty} E[h(\mathbf{w}(n), \mathbf{x}_1(n), \mathbf{x}_2(n))]$ is given by

$$\bar{\mathbf{h}}(\mathbf{w}, \mathbf{x}_1, \mathbf{x}_2) = \frac{\mathbf{w}(n+1) - \mathbf{w}(n)}{\eta} = \frac{d\mathbf{w}(t)}{dt} = \left(\frac{\mathbf{w}^T(t)\mathbf{R}_2\mathbf{w}(t)}{\mathbf{w}^T(t)\mathbf{R}_1\mathbf{w}(t)} \right) \mathbf{R}_2^{-1}\mathbf{R}_1\mathbf{w}(t) - \mathbf{w}(t) \quad \text{(C.8)}$$

where, η is the typical step-size parameter which is set to unity in this case. We want to find the stable stationary points of this ODE. Let $\mathbf{w}(t)$ be expanded in terms of the complete set of m generalized eigenvectors of $(\mathbf{R}_1, \mathbf{R}_2)$ as[4]

$$\mathbf{w}(t) = \sum_{k=1}^{m} \theta_k(t)\mathbf{q}_k \quad \text{(C.9)}$$

where, $\theta_k(t)$ is a time-varying projection and $\mathbf{q}_k$ is the generalized eigenvector corresponding to the eigenvalue λ_k. Using the simultaneous diagonalization property of the generalized eigenvectors, we can rewrite (C.8) using (C.9) as

$$\frac{d\theta_k(t)}{dt} = \frac{\sum_{l=1}^{m} \theta_l^2(t)}{\sum_{l=1}^{m} \lambda_l \theta_l^2(t)} \lambda_k \theta_k(t) - \theta_k(t) \quad \text{(C.10)}$$

We will analyze the dynamics of the non-linear differential equation in (C.10) separately.

[4] Any vector can be expressed as a linear combination of the complete set of basis vector spanning the vector space. In this case, since $\mathbf{w}(t)$ is operating in the joint space, it can be represented as a linear combination of the generalized eigenvectors spanning the space. Also note that the generalized eigenvectors are the principal components of the matrix $R^{-1}{}_2R_1$.

The goal is to show that the time varying projections corresponding to the modes associated with all eigenvectors except the principal eigenvector decay to zero asymptotically.For $1 < k \le m$, we define $\alpha_k(t) = \theta_k(t)/\theta_1(t)$. Therefore, by simple algebra, $\frac{d\alpha_k(t)}{dt} = \frac{1}{\theta_1(t)}\frac{d\theta_k(t)}{dt} - \frac{\alpha_k(t)}{\theta_1(t)}\frac{d\theta_1(t)}{dt}$, which can be further simplifed to

$$\frac{d\alpha_k(t)}{dt} = -\alpha_k(t)\left[\sum_{l=1}^{m}\theta_l^2(t)\Big/\sum_{l=1}^{m}\lambda_l\theta_l^2(t)\right]\{\lambda_1 - \lambda_k\} \tag{C.11}$$

$$d\alpha_k(t)/dt = -f(t)\alpha_k(t)(\lambda_1 - \lambda_k),\ f(t) = \sum_{l=1}^{m}\theta_l^2(t)\Big/\sum_{l=1}^{m}\lambda_l\theta_l^2(t) \tag{C.12}$$

Note that $f(t) > 0$ for all t. Therefore, it can be easily shown using Lyapunov stability theorems that, with $\lambda_1 > \lambda_2 > \lambda_3 > \lambda_4 > ....\lambda_m > 0, \alpha_k(t) = 0$ as $t \to \infty$, $k > 1$ and $\mathbf{w}(t) = \pm c\mathbf{q}_1$, where c is an arbitrary constant. As we are hard-limiting the weights to unity norm, eventually $\mathbf{w}(t) \to \mathbf{q}_1$. Thus, the principal generalized eigenvector is the stable stationary point of the ODE.

We will now prove that all other stationary points, i.e., the $m-1$ minor generalized eigenvectors are saddle points. Linearizing the ODE in (C.8) in the vicinity of a stationary point, we can compute the linearization matrix $\mathbf{A}$ as, $\mathbf{A} = \partial\left(\bar{\mathbf{h}}(\mathbf{w}(t))\right)/\partial\left(\mathbf{w}(t)\right)\Big|_{\mathbf{w}(t)=\mathbf{q}_k} = \mathbf{R}_2^{-1}\mathbf{R}_1\left(1/\lambda_k\right) - \mathbf{I}_{m\times m}$. The eigenvalues of the matrix $\mathbf{A}$ are given by $\lambda_A^m = \lambda_m/\lambda_k - 1$. It is easy to see that only for $k = 1$, all the eigenvalues which are analogous to the *s*-poles are within the LHP except the first pole which is at zero. All other stationary points have one or more poles in the RHP and hence they are saddle points. This means that near convergence, if the weight vector reaches any of the $m-1$ saddle points, it will diverge from that point and converge only to the stable stationary

point which is the principal generalized eigenvector. This completes the proof.

We would like to mention that the constant η in the ODE equation given by (C.8) is unity for our algorithm which makes this a constant gain algorithm. This constant essentially relates how well the discrete-time update equation approximates the ODE. When we make gain unity, we might be introducing discretization errors in our analysis which can lead to ambiguous results. This has been reported earlier in the PCA analysis and the trick is to analyze the discrete-time behavior. The *z*-poles can be extracted from the *s*-poles derived above using the transformation that we used for converting the update equation to the ODE. In this case, the transformation is simply given by $z = \eta s + 1$ with $\eta = 1$. The corresponding *z*-poles for the stable stationary point q_1 are given by λ_m / λ_1 for all values of m. Thus, we can easily deduce that the first pole is on the unit circle at $z = 1$ and all other z-poles reside inside the unit circle. Because of this pole at $z = 1$, the weight vector converges to a scaled value of the principal generalized eigenvector as shown before. A simple normalization will give us the exact result. Now, it suffices to say that even with the constant unity gain (step-size), the fixed-point update converges to the exact solution [75,76]. A smaller value of η can be used and this will strongly tie convergence of the ODE to that of the discrete-time update equation, but reduces the speed of convergence.

Theorem 2:.The weight vector $\mathbf{w}(n)$ enters a compact subset M of the basin of attraction $D(\mathbf{w}^*)$ infinitely often, with probability 1.

Proof: The domain of attraction $D(\mathbf{w}^*)$ includes all vectors with bounded norm. Also, let the initial weights are chosen such that $\mathbf{w}^T(0)\mathbf{q}_1 \neq 0$, where $\mathbf{q}_1$ is the principal generalized eigenvector (assumption 5). Let M be a compact subset defined by the set of

vectors with norm less than or equal to a finite constant. We are forcing the norm to be unity after every update. Thus, the weight vector $\mathbf{w}(n)$ will always lie inside the compact subset M.

From theorems 1 and 2, we can deduce from the theory of stochastic approximation for adaptive algorithms with constant gains [80] that, $\limsup_{t\to\infty} P\{\|\mathbf{w}(t)-\mathbf{q}_1\|>\in\}\le C(\eta)$, where $\in>0$ and $C(\eta)$ is a small constant which becomes zero when $\eta\to 0$.

The proposed GED algorithm has been successfully used to design optimum linear Multiuser Detectors [100-104] for Direct Sequence (DS) CDMA systems based on the receiver design proposed by Wong *et al* [76, 104, 105]. The same algorithm can be easily used to solve the Extended TLS problem discussed in Chapter 1. More details on the algorithm and the simulation results can be found in [76].

APPENDIX D
SOME DERIVATIONS FOR THE NOISY INPUT CASE

Consider the matrices **R**, **S**, **P**, and **Q** estimated from noisy data. For **R**, we write

$$\begin{aligned}\mathbf{R} &= E[\mathbf{x}(n)\mathbf{x}^T(n)] = E[(\tilde{\mathbf{x}}(n)+\mathbf{v}(n))(\tilde{\mathbf{x}}(n)+\mathbf{v}(n))^T] \\ &= E[\tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^T(n)+\tilde{\mathbf{x}}(n)\mathbf{v}^T(n)+\mathbf{v}(n)\mathbf{x}^T(n)+\mathbf{v}(n)\mathbf{v}^T(n)] \\ &= E[\tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^T(n)+\mathbf{v}(n)\mathbf{v}^T(n)] = \tilde{\mathbf{R}}+\mathbf{V}\end{aligned} \tag{D.1}$$

Similarly, for **S**, **P** and **Q** matrices, we obtain

$$\begin{aligned}\mathbf{S} &= E[\mathbf{x}(n)\mathbf{x}^T(n)+\mathbf{x}(n)\mathbf{x}^T(n)-\mathbf{x}(n)\mathbf{x}^T(n-L)-\mathbf{x}(n-L)\mathbf{x}^T(n)] \\ &= 2\mathbf{R}-E[(\tilde{\mathbf{x}}(n)+\mathbf{v}(n))(\tilde{\mathbf{x}}(n-L)+\mathbf{v}(n-L))^T+(\tilde{\mathbf{x}}(n-L)+\mathbf{v}(n-L))(\tilde{\mathbf{x}}(n)+\mathbf{v}(n))^T] \\ &= 2(\tilde{\mathbf{R}}+\mathbf{V})-E\begin{bmatrix}\tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^T(n-L)+\tilde{\mathbf{x}}(n)\mathbf{v}^T(n-L)+\mathbf{v}(n)\tilde{\mathbf{x}}^T(n-L)+\mathbf{v}(n)\mathbf{v}^T(n-L) \\ +\tilde{\mathbf{x}}(n-L)\tilde{\mathbf{x}}^T(n)+\tilde{\mathbf{x}}(n-L)\mathbf{v}^T(n)+\mathbf{v}(n-L)\tilde{\mathbf{x}}^T(n)+\mathbf{v}(n-L)\mathbf{v}^T(n)\end{bmatrix} \\ &= 2(\tilde{\mathbf{R}}+\mathbf{V})-E[\tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^T(n-L)+\tilde{\mathbf{x}}(n-L)\tilde{\mathbf{x}}^T(n)]-E[\mathbf{v}(n)\mathbf{v}^T(n-L)+\mathbf{v}(n-L)\mathbf{v}^T(n)] \\ &= 2(\tilde{\mathbf{R}}+\mathbf{V})-\tilde{\mathbf{R}}_L-\mathbf{V}_L\end{aligned} \tag{D.2}$$

$$\begin{aligned}\mathbf{P} &= E[\mathbf{x}(n)d(n)] = E[(\tilde{\mathbf{x}}(n)+\mathbf{v}(n))(\tilde{d}(n)+u(n))] \\ &= E[\tilde{\mathbf{x}}(n)\tilde{d}(n)+\tilde{\mathbf{x}}(n)u(n)+\mathbf{v}(n)\tilde{d}(n)+\mathbf{v}(n)u(n)] \\ &= E[\tilde{\mathbf{x}}(n)\tilde{d}(n)] = \tilde{\mathbf{P}}\end{aligned} \tag{D.3}$$

$$\begin{aligned}\mathbf{Q} &= E[(\mathbf{x}(n)-\mathbf{x}(n-L))(d(n)-d(n-L))] \\ &= E[\mathbf{x}(n)d(n)-\mathbf{x}(n)d(n-L)-\mathbf{x}(n-L)d(n)+\mathbf{x}(n-L)d(n-L)] \\ &= 2\mathbf{P}-E[\mathbf{x}(n)d(n-L)+\mathbf{x}(n-L)d(n)] \\ &= 2\tilde{\mathbf{P}}-E[(\tilde{\mathbf{x}}(n)+\mathbf{v}(n))(\tilde{d}(n-L)+u(n-L))+(\tilde{\mathbf{x}}(n-L)+\mathbf{v}(n-L))(\tilde{d}(n)+u(n))] \\ &= 2\tilde{\mathbf{P}}-E[\tilde{\mathbf{x}}(n)\tilde{d}(n-L)+\tilde{\mathbf{x}}(n-L)\tilde{d}(n)]-E\begin{bmatrix}\tilde{\mathbf{x}}(n)u(n-L)+\mathbf{v}(n)\tilde{d}(n-L) \\ +\mathbf{v}(n)u(n-L)+\tilde{\mathbf{x}}(n-L)u(n) \\ +\mathbf{v}(n-L)\tilde{d}(n)+\mathbf{v}(n-L)u(n)\end{bmatrix} \\ &= 2\tilde{\mathbf{P}}-\tilde{\mathbf{P}}_L\end{aligned} \tag{D.4}$$

APPENDIX E
ORTHOGONALITY OF ERROR TO INPUT

Recall that the optimal solution of AEC satisfies equation (2.9), which is equivalently

$$E[(1+2\beta)e_k\mathbf{x}_k - \beta(e_k\mathbf{x}_{k-L} + e_k\mathbf{x}_{k+L})] = 0 \tag{E.1}$$

Rearranging the terms in (E.1), we obtain

$$E[e_k(\mathbf{x}_k - \beta(\mathbf{x}_{k+L} - 2\mathbf{x}_k + \mathbf{x}_{k-L}))] = 0 \tag{E.2}$$

Notice that $(\mathbf{x}_{k+L} - 2\mathbf{x}_k + \mathbf{x}_{k-L})$ forms an estimate of the *acceleration* of the input vector $\mathbf{x}_k$. Specifically for $\beta = -1/2$, the term that multiplies e_k becomes a single-step prediction for the input vector $\mathbf{x}_k$ (assuming zero velocity and constant acceleration), according to Newtonian mechanics. Thus, the optimal solution of EWC tries to decorrelate the error signal from the predicted next input vector.

APPENDIX F
AEC AND ERROR ENTROPY MAXIMIZATION

This appendix aims to motivate an understanding of the relationship between entropy and sample differences. In general, the parametric family describing the error probability density function (pdf) in supervised learning is not analytically available. In such circumstances, non-parametric approaches such as Parzen windowing [115] could be employed. Given the i.i.d. samples {e(1),…,$e(N)$ } of a random variable e, the Parzen window estimate for the underlying pdf $f_e(.)$ is obtained by

$$\hat{f}_e(x) = \frac{1}{N}\sum_{i=1}^{N}\kappa_\sigma(x - e(i)) \tag{F.1}$$

where $\kappa_\sigma(.)$ is the kernel function, which itself is a pdf, and σ is the kernel size that controls the width of each *window*. Typically, Gaussian kernels are preferred, but other kernel functions like the Cauchy density [114] or the members of the generalized Gaussian family can be employed.

Shannon's entropy for a random variable e with pdf $f_e(.)$ is defined as

$$H(e) = -\int_{-\infty}^{\infty} f_e(x)\log f_e(x)dx = -E_e[f_e(e)] \tag{F.2}$$

Given i.i.d. samples, this entropy could be estimated [116] using

$$\hat{H}(e) = -\frac{1}{N}\sum_{j=1}^{N}\log\left(\frac{1}{N}\sum_{i=1}^{N}\kappa_\sigma(e(j) - e(i))\right) \tag{F.3}$$

This estimator uses the sample mean approximation for the expected value and the Parzen window estimator for the pdf. Viola proposed a similar entropy estimator, in

which he suggested dividing the samples into two subsets: one for estimating the pdf, the other for evaluating the sample mean [117]. In order to approximate a stochastic entropy estimator, we approximate the expectation by evaluating the argument at the most recent sample, e_k. In order to estimate the pdf, we use the L previous samples. The stochastic entropy estimator then becomes

$$\overline{H}(e) = -\log\left(\frac{1}{L}\sum_{i=1}^{L}\kappa_\sigma(e(k)-e(i))\right) \tag{F.4}$$

For supervised training of an ADALINE (or an FIR filter), with weight vector $\mathbf{w} \in \Re^n$, given the input (vector)-desired training sequence $(\mathbf{x}(n),d(n))$, where $\mathbf{x}(n) \in \Re^m$ and $d(n) \in \Re$, the instantaneous error is given by $e(n) = d(n) - \mathbf{w}^T(n)\mathbf{x}(n)$. The stochastic gradient of the error entropy with respect to the weights becomes

$$\frac{\partial \overline{H}(\mathbf{X})}{\partial \mathbf{w}} = -\left(\sum_{i=1}^{L}\kappa'_\sigma(e(n)-e(n-i))\big(\mathbf{x}(n)-\mathbf{x}(n-i)\big)\right)\Bigg/\sum_{i=1}^{L}\kappa_\sigma(e(n)-e(n-i)) \tag{F.5}$$

where $e(n-i) = d(n-i) - \mathbf{w}^T(n)\mathbf{x}(n-i)$ is also evaluated using the same weight vector as $e(n)$ [116]. For the specific choice of a single error sample $e(k\text{-}L)$ for pdf estimation and a Gaussian kernel function, (F.5) reduces to

$$\frac{\partial \overline{H}(\mathbf{X})}{\partial \mathbf{w}} = -(e(n)-e(n-L))\big(\mathbf{x}(n)-\mathbf{x}(n-L)\big)/\sigma^2 \tag{F.6}$$

We easily notice that the expression in (F.6) is also a stochastic gradient for the cost function $J(\mathbf{w}) = E[(e(n)-e(n-L))^2]/(2\sigma^2)$, which is essentially a scaled form of the second term in the AEC.

APPENDIX G
PROOF OF CONVERGENCE OF ERROR VECTOR NORM IN AEC-LMS

The dynamics of the error vector norm is given by

$$\left\|\boldsymbol{\varepsilon}_{k+1}\right\|^2 = \left\|\boldsymbol{\varepsilon}_k\right\|^2 - 2\eta sign(e_k^2 + \beta\dot{e}_k^2)\boldsymbol{\varepsilon}_k^T(e_k\mathbf{x}_k + \beta\dot{e}_k\dot{\mathbf{x}}_k) + \eta^2\left\|e_k\mathbf{x}_k + \beta\dot{e}_k\dot{\mathbf{x}}_k\right\|^2 \tag{G.1}$$

Further, since $\boldsymbol{\varepsilon}_k^T\mathbf{x}_k = e_k$ and $\boldsymbol{\varepsilon}_k^T\dot{\mathbf{x}}_k = \dot{e}_k$, we have

$$\left\|\boldsymbol{\varepsilon}_{k+1}\right\|^2 = \left\|\boldsymbol{\varepsilon}_k\right\|^2 - 2\eta\left|e_k^2 + \beta\dot{e}_k^2\right| + \eta^2\left\|e_k\mathbf{x}_k + \beta\dot{e}_k\dot{\mathbf{x}}_k\right\|^2 \tag{G.2}$$

Define the following term.

$$\varphi = 2 - \eta\frac{\left\|e_k\mathbf{x}_k + \beta\dot{e}_k\dot{\mathbf{x}}_k\right\|^2}{\left|e_k^{\,2} + \beta\dot{e}_k^2\right|} \tag{G.3}$$

If the step-size (positive) upper bound in equation (4.13) is satisfied, then $\varphi > 0$ for all *k*. Therefore, equation G.3 reduces to the inequality

$$\left\|\boldsymbol{\varepsilon}_{k+1}\right\|^2 \le \left\|\boldsymbol{\varepsilon}_k\right\|^2 - \eta\varphi\left|e_k^2 + \beta\dot{e}_k^2\right| \tag{G.4}$$

Iterating (G.4) from *k* = 0, we get, $\left\|\boldsymbol{\varepsilon}_k\right\|^2 \le \left\|\boldsymbol{\varepsilon}_0\right\|^2 - \eta\varphi\sum_{t=1}^{k}\left|e_t^2 + \beta\dot{e}_t^2\right|$. In the limit $k \to \infty$, it is easy to see that, $\left\|\boldsymbol{\varepsilon}_\infty\right\|^2 + \varphi\eta\sum_{t=1}^{\infty}\left|e_t^2 + \beta\dot{e}_t^2\right| \le \left\|\boldsymbol{\varepsilon}_0\right\|^2$ which implies that $\lim_{t\to\infty}\left|e_t^2 + \beta\dot{e}_t^2\right| = 0$ as the summation in the error terms must converge to a finite value given by $\left\|\boldsymbol{\varepsilon}_0\right\|^2 - \left\|\boldsymbol{\varepsilon}_\infty\right\|^2$. The instantaneous cost $\left|e_t^2 + \beta\dot{e}_t^2\right|$ becomes zero only when the weights converge to the true weights $\mathbf{w}_T$ ($\therefore \left\|\boldsymbol{\varepsilon}_\infty\right\|^2 = 0$). Also note that the gradient becomes zero at this point.

LIST OF REFERENCES

1. Hoffman, K., Kunze, R. *Linear Algebra*. Prentice-Hall, New Delhi, India, 1996.

2. Principe, J.C., Euliano, N., Lefebvre, C. *Neural and Adaptive Systems: Fundamentals Through Simulations*. John Wiley, New York, 2000.

3. Hestenes, M.R. *Optimization Theory*. John Wiley, New York, 1975.

4. Scharf, L.L. *Statistical Signal Processing*. Addison-Wesley, Boston, MA, 1991.

5. Orfanidis, S.J. *Optimum Signal Processing*: *An Introduction*. McGraw-Hill, Singapore, 1990.

6. Wiener, N. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series with Engineering Applications*. MIT Press, Cambridge, MA, 1949.

7. Meyer, C.D. *Matrix Analysis and Applied Linear Algebra*. SIAM, Philadelphia, PA, 2001.

8. Golub, G.H., van Loan, C.F. *Matrix Computations*. The John Hopkins University Press, London, UK, 1996.

9. Widrow, B., Hoff, Jr. M.E. "Adaptive Switching Circuits," Proceedings of IRE WESCON Convention Record, part 4, pp. 96–104, 1960.

10. Widrow, B., Stearns, S. *Adaptive Signal Processing*. Prentice-Hall, Englewood Cliffs, NJ, 1985.

11. Macchi, O. *Adaptive Processing*: *The Least Mean Squares Approach with Applications in Transmission*. John Wiley, New York, 1995.

12. Haykin, S., Widrow, B (eds.). *Least-Mean-Square Adaptive Filters*. John Wiley, New York, 2003.

13. Solo, V., Kong, X. *Adaptive Signal Processing Algorithms*: *Stability and Performance*. Prentice-Hall, Englewood Cliffs, NJ, 1995.

14. Haykin, S. *Adaptive Filter Theory*. Prentice Hall, Upper Saddle River, NJ, 1996.

15. Farhang-Boroujeny, B. *Adaptive Filters*: *Theory and Applications*. John Wiley, New York, 1998.

16. Luenberger, D. *Optimization by Vector Space Methods*. John Wiley, New York, 1969.

17. Rao, S.S. *Engineering Optimization*: *Theory and Practice*. John Wiley, New Delhi, 1996.

18. Kalman, R.E. "New methods in Wiener filtering theory," Research Institute for Advanced Studies, Rep. 61-1, Baltimore, MD, 1961.

19. Mueller, M. "Least-Squares Algorithms for Adaptive Equalizers," Bell Systems Technical Journal, vol. 60, pp. 1905-1925, 1981.

20. Lucky, R.W. "Techniques for Adaptive Equalization of Digital Communications Systems," Bell Systems Technical Journal, vol. 45, pp. 255-286, 1966.

21. Verhoeckx, N.A.M., van den Elzen, H.C., Snijders, F.A.M., van Gerwen, P.J. "Digital Echo Cancellation for Baseband Data Transmission," IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 27, pp. 761–781, June 1979.

22. Jayant, N.S., Noll, P. *Digital Coding of Waveforms*. Prentice-Hall, Englewood Cliffs, NJ, 1984.

23. Widrow, B., McCool, J.M., Larimore, M.G., Johnson, C.R. "Stationary and Nonstationary Learning Characteristic of the LMS Adaptive Filter," Proceedings of the IEEE, vol. 64, pp. 1151–1162, 1976.

24. Harris, R., Chabries, D., Bishop, F.A. "A Variable Step (VS) Adaptive Filter Algorithm," IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 34, pp. 309–316, April 1986.

25. Kwong, R., Johnston, E.W. "A Variable Step Size LMS Algorithm," IEEE Transactions on Signal Processing, vol. 40, pp. 1633–1642, July 1992.

26. Aboulnasr, T., Mayyas, K. "A Robust Variable Step-size LMS-type Algorithm: Analysis and Simulations," IEEE Transactions on Signal Processing, vol. 45, pp. 631–639, March 1997.

27. Wei, Y., Gelfand, S., Krogmeier, J.V. "Noise-Constrained Least Mean Squares Algorithm," IEEE Transactions on Signal Processing, vol. 49, no. 9, pp. 1961-1970, September 2001.

28. Sethares, W., Lawrence, D., Johnson, Jr. C., Bitmead, R. "Parameter Drift in LMS Adaptive Filters," IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 34-4, pp. 868–879, August 1986.

29. Liavas, A., Regalia, P. "On the Numerical Stability and Accuracy of the Conventional Recursive Least Squares Algorithm," IEEE Transactions on Signal Processing, vol. 47, no. 1, pp. 88-96, January 1999.

30. Slock, D.T.M., Kailath, T. “Numerically Stable Fast Transversal Filters for Recursive Least Squares Adaptive Filtering,” IEEE Transactions on Signal Processing, vol. 39, pp. 92–114, January 1991.

31. Eleftheriou, E., Falconer, D. “Tracking Properties and Steady-state Performance of RLS Adaptive Filter Algorithms,” IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 34, pp. 1097–1109, October 1986.

32. Söderström, T., Stoica, P. *System Identification*. Prentice-Hall, London, UK, 1989.

33. Erdogmus, D., Principe, J.C. “An Error-Entropy Minimization Algorithm for Supervised Training of Nonlinear Adaptive Systems,” IEEE Transactions on Signal Processing, vol. 50, no. 7, pp. 1780-1786, July 2002.

34. Principe, J.C., Fisher, J. III., Xu, D. “Information theoretic learning,” in Haykin, S. (ed), *Unsupervised Adaptive Filtering*. John Wiley, New York, 2000.

35. Cadzow, J.A. “Total Least Squares, Matrix Enhancement, and Signal Processing,” *Digital Signal Processing*, vol. 4, pp. 21-39, 1994.

36. de Moor, B. “Total Least Squares for Affinely Structured Matrices and the Noisy Realization Problem,” IEEE Transactions on Signal Processing, vol. 42, pp. 3104-3113, November 1994.

37. Lemmerling, P. “Structured Total Least Squares: Analysis, Algorithms, and Applications,” Ph.D. Dissertation, Katholeike University, Leuven, Belgium, 1999.

38. Yeredor, A. “The Extended Least Squares Criterion: Minimization Algorithms and Applications,” IEEE Transactions on Signal Processing, vol. 49, no. 1, pp. 74-86, January, 2001.

39. Feng, D.Z., Bao, Z., Jiao, L.C. “Total Least Mean Squares Algorithm,” IEEE Transactions on Signal Processing, vol. 46, no. 8, pp. 2122-2130, August 1998.

40. Davila, C.E. “An Efficient Recursive Total Least Squares Algorithm for FIR Adaptive Filtering.” IEEE Transactions on Signal Processing, vol. 42, no. 2, pp. 268-280, February 1994.

41. Deprettere, F. (ed.). *SVD and Signal Processing – Algorithms, Applications and Architectures*. North-Holland, Amsterdam, 1988.

42. Cichocki A., Amari, S. *Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications*. John Wiley, New York, 2002.

43. Ljung, L., Söderström, T. *Theory and Practice of Recursive System Identification*. MIT Press, Cambridge, MA, 1983.

44. Rao, Y.N. “Algorithms for Eigendecomposition and Time Series Segmentation,” MS Thesis, University of Florida, Gainesville, FL, 2000.

45. Davila, C.E. “A Subspace Approach to Estimation of Autoregressive Parameters from Noisy Measurements,” IEEE Transactions on Signal Processing, vol. 46, no. 2, pp. 531-534, February, 1998.

46. So, H.C. “Modified LMS Algorithm for Unbiased Impulse Response Estimation in Nonstationary Noise,” Electronics Letters, vol. 35, no. 10, pp. 791-792, May 1999.

47. Gao, K., Ahmad, M.O., Swamy, M.N.S. “A Constrained Anti-Hebbian Learning Algorithm for Total Least Squares Estimation with Applications to Adaptive FIR and IIR Filtering,” IEEE Transactions on Circuits and Systems-Part II, vol. 41, no. 11, pp. 718-729, November 1994.

48. Mathew, G., Reddy, V. U., Dasgupta, S. “Adaptive Estimation of Eigensubspace,” IEEE Transactions on Signal Processing, vol. 43, no. 2, pp. 401-411, February 1995.

49. Zhang, Q., Leung, Y. “A Class of Learning Algorithms for Principal Component Analysis and Minor Component Analysis,” IEEE Transactions on Neural Networks, vol. 11, no. 1, pp. 200-204 January, 2000.

50. Luo, F. L., Unbehauen, R. “A Minor Subspace Analysis Algorithm,” IEEE Transactions on Neural Networks, vol. 8, no. 5, pp. 1149-1155, September 1997.

51. Xu, L., Oja, E., Suen, C. Y. “Modified Hebbian learning for Curve and Surface Fitting,” Neural Networks, vol. 5, pp. 441-457, 1992.

52. Joliffe, I.T. *Principal Component Analysis*. Springer-Verlag, Berlin, 1986.

53. Duda, R.O., Hart, P.E. *Pattern Classification and Scene Analysis*. John Wiley, New York, 1973.

54. Kung, S.Y., Diamantaras, K.I., Taur, J.S. “Adaptive Principal Component Extraction (APEX) and Applications,” IEEE Transactions on Signal Processing, vol. 42, no. 5, pp. 1202-1217, May 1994.

55. Mao, J., Jain, A.K. “Artificial Neural Networks for Feature Extraction and Multivariate Data Projection,” IEEE Transactions on Neural Networks, vol. 6, no. 2, pp. 296-317, March 1995.

56. Sanger, T.D. “Optimal Unsupervised Learning in a Single-layer Linear Feedforward Neural Network,” Neural Networks, vol. 2, pp. 459-473, 1989.

57. Rubner, J., Tavan, P. “A Self-Organizing Network for Principal Component Analysis,” Europhysics Letters, vol. 10, no. 7, pp. 693-698, 1989.

58. Rubner, J., Tavan, P. “Development of Feature Detectors by Self Organization,” Biological Cybernetics, vol. 62, pp. 193-199, 1990.

59. Diamantaras, K.I., Kung, S.Y. *Principal Component Neural Networks, Theory and Applications*. John Wiley, New York, 1996.

60. Hua, Y., Xiang, Y., Chen, T., Abed-Meraim, K., Miao, Y. “Natural Power Method for Fast Subspace Tracking,” Proceedings of the 1999 IEEE Signal Processing Society Workshop, pp. 176-185, August 1999.

61. Yang, B. “Projection Approximation Subspace Tracking,” IEEE Transactions on Signal Processing, vol. 43, no. 1, pp. 95-107, January 1995.

62. Oja, E. “A Simplified Neuron Model as a Principal Component Analyzer,” Journal of Mathematical Biology, vol.15, pp. 267-273, 1982.

63. Baldi, P., Hornik, K. “Learning in Linear Neural Networks: A Survey,” IEEE Transactions on Neural Networks, vol. 6, no. 4, pp. 837-858, July 1995.

64. Xu, L. “Least Mean Square Error Reconstruction Principle for Self-Organizing Neural Nets,” Neural Networks, vol. 6, pp. 627-648, 1993.

65. Chatterjee, C., Kang, Z., Roychowdhury, V.P. “Algorithms for Accelerated Convergence of Adaptive PCA”. IEEE Transactions on Neural Networks, vol. 11, no. 2, pp. 338-355, March 2000.

66. Erdogmus, D., Rao, Y.N., Principe, J.C., Fontenla-Romero, O., Vielva, L. “An Efficient, Robust and Fast Converging Principal Components Extraction Algorithm: SIPEX-G,” Proceedings of European Signal Processing Conference, vol. 2, pp. 335-338, September 2002.

67. Erdogmus, D., Rao, Y.N., Ozturk, M.C., Vielva, L., Principe, J.C. “On the Convergence of SIPEX: A Simultaneous Principal Components Extraction Algorithm,” Proceedings of International Conference on Acoustics, Speech and Signal Processing, vol. 2, pp. 697-700, April 2003.

68. Erdogmus, D., Rao, Y.N., Hild, K.E., Principe, J.C. “Simultaneous Principal Component Extraction with Application to Adaptive Blind Multiuser Detection,” EURASIP Journal on Applied Signal Processing, vol. 2002, no. 12, pp. 1473-1484, December 2002.

69. Chatterjee, C., Roychowdhury, V.P., Ramos, J., Zoltowski, M.D. “Self-Organizing Algorithms for Generalized Eigendecomposition,” IEEE Transactions on Neural Networks, vol. 8, no. 6, pp. 1518-1530, November 1997.

70. Miao, Y., Hua, Y. “Fast Subspace Tracking and Neural Network Learning by a Novel Information Criterion,” IEEE Transactions on Signal Processing, vol. 46 no. 7, pp. 1967-1979, July 1998.

71. Rao, Y.N, Principe, J.C. "A Fast On-line Generalized Eigendecomposition Algorithm for Time Series Segmentation," Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000, pp. 266-271, October 2000.

72. Rao, Y.N., Principe, J.C. "A Fast On-line Algorithm for PCA and its Convergence Characteristics," Proceedings of the 2000 IEEE Signal Processing Society Workshop , vol. 1 , pp. 299-307, December 2000.

73. Rao, Y.N., Principe, J.C. "Robust On-line Principal Component Analysis Based on a Fixed-Point Approach," Proceedings of International Conference on Acoustics, Speech and Signal Processing, vol. 1, pp. 981-984, May 2002.

74. Rao, Y.N., Principe, J.C., "Time Series Segmentation Using a Novel Adaptive Eigendecomposition Algorithm," Journal of VLSI Signal Processing, vol. 32, pp. 7-17, 2002.

75. Rao, Y.N., Principe, J.C. "An RLS Type Algorithm for Generalized Eigendecomposition," Proceedings of the 2001 IEEE Signal Processing Society Workshop**,** pp. 263-272, September 2001.

76. Rao, Y.N., Principe, J.C., Wong, T.F. "Fast RLS-like Algorithm for Generalized Eigendecomposition and its Applications," Journal of VLSI Signal Processing, vol. 37, pp. 333-344, 2004.

77. Rao, Y.N. "Optimal Adaptive Projections Using Stochastic and Recursive Algorithms and their Applications," Ph.D. proposal, University of Florida, Gainesville, FL, December 2002.

78. Ljung, L. "Analysis of Recursive Stochastic Algorithms," IEEE Transactions on Automatic Control, vol. 22, no. 4, pp. 551-575, August 1977.

79. Kushner, H.J., Clark, D.S. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*. Springer-Verlag, New York, 1978.

80. Benveniste, A., Metivier, M., Priouret, P. *Adaptive Algorithms and Stochastic Approximations*. Springer-Verlag, Berlin, 1990.

81. Kushner, H.J., Yin, G. *Stochastic Approximation Algorithms and Applications*. Springer-Verlag, Berlin 1997.

82. Plumbley, M.D. "Lyaponuv Functions for Convergence of Principal Component Algorithms," Neural Networks, vol. 8, no. 1, 1995.

83. Proakis, J.G. *Digital Communications*. McGraw-Hill, New York, 2001.

84. Haykin, S. *Neural Networks*: *A Comprehensive Foundation*. Prentice-Hall, Englewood Cliffs, NJ, 1999.

85. Moon, T.K., Stirling, W.C. *Mathematical Methods and Algorithms for Signal Processing*. Prentice-Hall, Englewood Cliffs, NJ, 1999.

86. Rao, Y.N., Principe, J.C. "Efficient Total Least Squares Method for System Modeling using Minor Component Analysis,". Proceedings of the IEEE Workshop on Neural Networks for Signal Processing XII, pp. 259-268, September 2002.

87. van Huffel, S., Vanderwalle, J. *The Total Least Squares Problem*: *Computational Aspects and Analysis*. SIAM, Philadelphia, PA, 1991.

88. Mathews, J., Cichocki, A. "Total Least Squares Estimation," Technical Report, University of Utah, USA and Brain Science Institute Riken, Japan, 2000.

89. Shynk, J.J. "Adaptive IIR Filtering," IEEE Signal Processing Magazine, vol. 6, pp. 4-21, April 1989.

90. Regalia, P.A. *Adaptive IIR Filtering in Signal Processing and Control*. Marcel Dekker, New York, 1995.

91. Regalia, P.A., "An Unbiased Equation Error Identifier and Reduced Order Approximations," IEEE Transactions on Signal Processing, vol. 42, no. 6, pp. 1397-1412, June 1994.

92. Regalia, P.A., "An Adaptive Unit Norm Filter with Applications to Signal Analysis and Karhunen-Loeve Transformations," IEEE Transactions on Circuits and Systems, vol. 37, no. 5, pp. 646-649, May 1990.

93. Söderström, T., Stoica, P. *Instrumental Variable Methods for System Identification*. Springer-Verlag, Berlin, 1983.

94. Fukunaga, K. *Introduction to Statistical Pattern Recognition*. Academic Press, New York, 1990.

95. Xu, D., Principe, J.C., Wu, H.C. "Generalized Eigendecomposition with an On-line Local Algorithm," IEEE Signal Processing Letters, vol. 5, no. 11, pp. 298-301, November 1998.

96. Mathew G., Reddy, V.U. "A Quasi-Newton Adaptive Algorithm for Generalized Symmetric Eigenvalue Problem," IEEE Transactions on Signal Processing, vol. 44, no. 10, pp. 2413-2422, October 1996.

97. Diamantaras, K.I., Kung, S.Y. "An Unsupervised Neural Model for Oriented Principal Component Extraction," Proceedings of International Conference on Acoustics, Speech and Signal Processing, pp. 1049-1052, vol. 2, May 1991.

98. Cao, Y., Sridharan, S., Moody, M. "Multichannel Speech Separation by Eigendecomposition and its Application to Co-talker Interference Removal," IEEE Transactions on Speech and Audio Processing, vol. 5, no. 3, pp. 209-219, May 1997.

99. Hyvärinen, A. "Fast and Robust Fixed-Point Algorithms for Independent Component Analysis," IEEE Transactions on Neural Networks, vol. 10, no. 3, pp. 626-634, May 1999.

100. Wang, X., Poor, H.V. "Blind Multiuser Detection: A Subspace Approach," IEEE Transactions on Information Theory, vol. 44, no. 2, pp. 677-691, March 1998.

101. Wang, X., Host-Madsen, A. "Group-Blind Multiuser Detection for Uplink CDMA," IEEE Journal on Selected Areas in Communications, vol. 17, no. 11, pp. 1971-1984, November 1999.

102. Reynolds, D., Wang, X. "Adaptive Group-Blind Multiuser Detection Based on a New Subspace Tracking Algorithm," IEEE Transactions on Communications, vol. 49, no. 7, pp. 1135-1141, July 2001.

103. Honig, M., Madhow, U., Verdu, S. "Blind Adaptive Multiuser Detection," IEEE Transactions on Information Theory, vol. 41, no. 4, pp. 944-960, July 1995.

104. Wong, T.F., Lok, T.M., Lehnert, J.S., Zoltowski, M.D. "A Linear Receiver for Direct-Sequence Spread-Spectrum Multiple-Access Systems with Antenna Arrays and Blind Adaptation", IEEE Transactions on Information Theory, vol. 44, no. 2, pp. 659-676, March 1998.

105. Rao, Y.N. "Linear Receiver for Direct-Sequence Spread-Spectrum Multiple Access Systems," CNEL technical report, December 2001.

106. Hahn, S. L. *Hilbert Transforms in Signal Processing*. Artech House, London, UK, 1996.

107. Shannon, C.E., Weaver, W. *The Mathematical Theory of Communication*. University of Illinois Press, Urbana, IL, 1964.

108. Tarasenko, F.P. "On the Evaluation of an Unknown Probability Density Function, the Direct Estimation of the Entropy from Independent Observations of a Continuous Random Variable, and the Distribution-Free Entropy Test of Goodness-of-fit," Proceedings of IEEE, vol. 56, pp. 2052-2053, 1968.

109. Bickel, P.J., Breiman, L. "Sums of Functions of Nearest Neighbor Distances, Moment Bounds, Limit Theorems and a Goodness-of-fit Test," Annals of Statistics, vol. 11, no. 1, pp. 185-214, 1983.

110. Beirlant, J., Zuijlen, M.C.A. "The Empirical Distribution Function and Strong Laws for Functions of Order Statistics of Uniform Spacings," Journal of Multivariate Analysis, vol. 16, pp. 300-317, 1985.

111. Kozachenko, L.F., Leonenko, N.N. "Sample Estimate of Entropy of a Random Vector," Problems of Information Transmission, vol. 23, no. 2, pp. 95-101, 1987.

112. Beck, C., Schlogl, F. *Thermodynamics of Chaotic Systems*. Cambridge University Press, Cambridge, UK, 1993.

113. Tsybakov, A.B., van der Meulen, E.C. "Root-n Consistent Estimators of Entropy for Densities with Unbounded Support," *Scandinavian Journal of Statistics*, vol. 23, pp. 75-83, 1996.

114. Papoulis, A., Pillai, S.U. *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, New York, 2002.

115. Parzen, E. "On Estimation of a Probability Density Function and Mode," *Time Series Analysis Papers*, Holden-Day, Inc., San Diego, CA, 1967.

116. Erdogmus, D. "Information Theoretic Learning: Renyi's Entropy and its Applications to Adaptive System Training," Ph.D. Dissertation, University of Florida, Gainesville, FL, 2002.

117. Viola, P., Schraudolph, N., Sejnowski, T. "Empirical Entropy Manipulation for Real-World Problems," Proceedings of Neural Information Processing Systems, pp. 851-857, November 1995.

118. Bishop, C. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, UK, 1995.

119. Akaike, H. "A New Look at the Statistical Model Identification," IEEE Transactions on. Automatic Control, vol. 19, pp. 716-723, December 1974.

120. Rissanen, J. *Stochastic Complexity in Statistical Inquiry*. World Scientific, London, UK, 1989.

121. Principe, J.C., Rao, Y.N., Erdogmus, D. "Error Whitening Wiener Filters: Theory and Algorithms," in *Least-Mean-Square Adaptive Filters*, Haykin, S., Widrow, B. (eds.), John Wiley, New York, 2003.

122. Reiersøl, O. "Confluence Analysis by Means of Lag Moments and Other Methods of Confluence Analysis," Econometrica, vol. 9, pp. 1-23, 1941.

123. Wong, K.Y., Polak, E. "Identification of Linear Discrete Time Systems Using the Instrumental Variable Approach," IEEE Transactions on Automatic Control, vol. 12, no. 8, pp. 707-718, December 1967.

124. Young, P.C. “An Instrumental Variable Method for Real-Time Identification of a Noisy Process,” Automatica, vol. 6, no.2, pp. 271-287, March 1970.

125. Young, P.C. “Parameter Estimation for Continuous-Time Models: A Survey,” Automatica, vol. 17, no. 1, pp. 23-39, January 1981.

126. Young, P.C. *Recursive Estimation and Time Series Analysis*. Springer-Verlag, Berlin, 1984.

127. Rao, Y.N., Erdogmus, D., Rao, G.Y., Principe, J.C. “Fast Error Whitening Algorithms for System Identification and Control,” Proceedings of the IEEE Workshop of Neural Networks for Signal Processing, pp. 309-318, September 2003.

128. Rao, Y.N., Erdogmus, D., Principe, J.C. “Error Whitening Criterion for Adaptive Filtering: Theory and Algorithms,” to appear in IEEE Transactions on Signal Processing, 2005.

129. Chansarkar, M., Desai, U.B. “A Robust Recursive Least Squares Algorithm,” IEEE Transactions on Signal Processing, vol. 45, no. 7, pp. 1726-1735, July 1997.

130. Boyd, S., Vandenberghe, L. *Convex Optimization*. Lecture Notes, Stanford University, Winter 2001.

131. Al-Naffouri, T.Y., Sayed, A.H. “Adaptive Filters with Error Nonlinearities: Mean-Square Analysis and Optimum Design,” EURASIP Journal on Applied Signal Processing, no. 4, pp. 192-205, 2001.

132. Sayed, A.H. “Energy Conservation and the Learning Ability of LMS Adaptive Filters,” in *Least-Mean-Square Adaptive Filters*, Haykin, S., Widrow, B. (eds.), John Wiley, New York, 2003.

133. Price, R. “A Useful Theorem for Nonlinear Devices Having Gaussian Inputs,” IRE. Transactions on Information Theory, vol. 4, pp. 69-72, June 1958.

134. Eweda, E. “Convergence Analysis of the Sign Algorithm without the Independence and Gaussian Assumptions,” IEEE Transactions on Signal Processing, vol. 48, no. 9, pp. 2535-2544, September 2000.

135. Reuter, M., Quirk, K., Zeidler, J., Milstein, L. “Non-Linear Effects in LMS Adaptive Filters,” Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000, pp. 141-146, October 2000.

136. Widrow, B., Walach, E. *Adaptive Inverse Control*. Prentice-Hall, Englewood Cliffs, NJ, 1995.

137. Rao, Y.N., Erdogmus, D., Rao, G.Y., Principe, J.C, "Stochastic Error Whitening Algorithm for Linear Filter Estimation with Noisy Data," Neural Networks, vol. 16, no. 5-6, pp. 873-880, June 2003.

138. Rao, Y.N., Erdogmus, D., Principe, J.C.,"Error Whitening Criterion for Linear Filter Estimation," Proceedings of International Joint Conference on Neural Networks, vol. 2, pp. 1447-1452, July 2003.

139. Rao, Y.N., Erdogmus, D., Principe, J.C. "Error Whitening Methodology for Linear Parameter Estimation in Noisy (White Noise) Inputs," US patent, submitted, March 2003.

140. Robbins, H., Munro, S. "A stochastic optimization method," Annals of Mathematical Statistics, vol. 22, pp. 400-407, 1951.

141. Rao, Y.N., Erdogmus, D., Principe, J.C. "Accurate Linear Parameter Estimation in Colored Noise," accepted for publication in International Conference of Acoustics, Speech and Signal Processing, 2004.

142. Douglas, S.C., Rupp, M. "On bias removal and unit-norm constraints in equation-error adaptive filters," 30th Annual Asilomar Conference on Signals, Systems and Computers, CA, vol. 2, pp. 1093-1097, November 1996.

143. Rao, Y.N., Kim, S.P., Sanchez, J.C., Erdogmus, D., Principe, J.C., Carmena, J.M., Lebedev, M.A., Nicolelis, M.A.L. "Learning Mappings in Brain Machine Interfaces with Echo State Networks," accepted for publication in International Joint Conference on Neural Networks, 2004.

144. Cherkassky, V., Mulier, F. *Learning from Data*. John Wiley, New York, 1998.

145. Hastie, T., Tibshirani, R., Friedman, J. *The Elements of Statistical Learning*: *Data Mining, Inference, and Prediction*. Springer-Verlag, Berlin, 2001.

146. Chen, L., Narendra, K.S. "Nonlinear Adaptive Control Using Neural Networks and Multiple Models," Automatica, vol. 37, no. 8, pp. 1245-1255, August 2001.

147. Principe, J.C., Wang, L., Motter, M.A. "Local Dynamic Modeling with Self-Organizing Maps and Applications to Nonlinear System Identification and Control," Proceedings of the IEEE, vol. 86, no. 11, pp. 2240-2258, November 1998.

148. Thampi, G.K., Principe, J.C., Motter, M.A. Cho, J., Lan, J. "Multiple model based flight control design," Proceedings of the 45th Midwest Symposium on Circuits and Systems, vol. 3, pp. 133-136, August 2002.

149. Kim, S.P., Sanchez, J.C., Erdogmus, D., Rao, Y.N., Principe, J.C., Nicolelis, M.A.L. “Divide-and-Conquer Approach for Brain Machine Interfaces: Nonlinear Mixture of Competitive Linear Models,” Neural Networks, vol. 16, no. 5-6, pp. 865-871, Jun 2003.

150. Cho, J., Lan, J., Thampi, G.K., Principe, J.C., Motter, M.A. “Identification of Aircraft Dynamics Using a SOM and Local Linear Models,” Proceedings of the 45th Midwest Symposium on Circuits and Systems, vol. 2, pp. 148-151, August 2002.

BIOGRAPHICAL SKETCH

Yadunandana Nagaraja Rao was born in Mysore, India, on January 11, 1976. He graduated with a bachelor's degree in electronics and communication engineering from the University of Mysore in August 1997 and then worked as a software engineer at IT Solutions Inc., Bangalore, India, till July 1998. In fall 1998, he began graduate studies in the Department of Electrical and Computer Engineering at the University of Florida, Gainesville, FL. Yadu joined the Computational NeuroEngineering Laboratory (CNEL) in the spring of 1999 and obtained his master's degree (thesis option) in spring 2000 under the supervision of Dr. Jose C. Principe. After a brief stint as a design engineer at GE Medical Systems, WI, Yadu returned to CNEL in spring 2001 as a doctoral candidate. Since then, he has been working towards his Ph.D. in the Department of Electrical and Computer Engineering, under the able guidance of Dr. Jose C. Principe. His primary research interests include algorithm development, analysis, design of optimal adaptive learning mechanisms and neural networks. He is a member of the International Neural Network Society (INNS) and also a student member of the IEEE.