

ADAPTIVE STEP SIZE AND EXPONENTIALLY WEIGHTED AFFINE PROJECTION  
ALGORITHMS

by

Murat Çabuk

B.S. in E.E., Boğaziçi University, 1998

Submitted to the Institute for Graduate Studies in  
Science and Engineering in partial fulfillment of  
the requirements for the degree of  
Master of Science  
in  
Electrical and Electronics Engineering

Boğaziçi University

2002

ADAPTIVE STEP SIZE AND EXPONENTIALLY WEIGHTED AFFINE PROJECTION  
ALGORITHMS

APPROVED BY:

Assoc. Prof. Ayşın B. Ertüzün .....  
(Thesis Supervisor)

Prof. Emin Anarım .....  
(Thesis Co-Supervisor)

Assoc. Prof. Levent Arslan .....

Prof. Ahmet H. Kayran .....

Prof. Avni Morgül .....

DATE OF APPROVAL .....

## **ACKNOWLEDGEMENTS**

I would like to express my appreciation to my thesis supervisors Assoc. Prof. Ayşın B. Ertüzün and Prof. Emin Anarım for their excellent guidance, suggestions, contributions and encouragement throughout this study.

This work would not have been possible without the support and love of my family. I would also like to thank my family.

## **ABSTRACT**

### **ADAPTIVE STEP SIZE AND EXPONENTIALLY WEIGHTED AFFINE PROJECTION ALGORITHMS**

In the thesis, algorithms having both fast convergence and minimum misadjustment are proposed. Hence, we obtain algorithms that do not have convergence speed – misadjustment trade off problem. Affine Projection Algorithm (APA) is chosen for its fast convergence, and Adaptive Step Size (ASS) methods are proposed for APA. Additionally, Exponentially Weighted APA and its switching version are proposed.

To demonstrate performance improvements provided by the algorithms, some adaptive transversal filtering applications, namely system identification, acoustic echo cancellation and inverse system identification, are simulated.

## ÖZET

### UYARLAMALI BASAMAK ADIMLI VE ÜSTEL AĞIRLIKLI İLGİN İZDÜŞÜM ALGORİTMALARI

Tezde, hızlı yakınsayan ve en iyi yatışkın hal başarımını sağlayan algoritmalar önerilmiştir. Böylece, yakınsama hızı - yatışkın hal sapması ikilem problemi olmayan algoritmalar elde edilmiştir. İlgİN İzdüşüm Algoritması hızlı yakınsaması dolayısıyla seçilmiştir, ve bu algoritma için uyarlanırlar adım yöntemleri önerilmiştir. Ayrıca, Üstel Ağırlıklı İlgİN İzdüşüm Algoritması ve onun deęişmeli versiyonu da önerilmiştir.

Algoritmaların getirdiđi başarımı göstermek için bazı uyarlanırlar süzgeçleme uygulamaları için bilgisayar benzetimleri yürütülmüştür. Bu uygulamalar sistem modelleme, akustik yankı giderimi ve ters sistem modellemesidir.

## TABLE OF CONTENTS

ACKNOWLEDGEMENTS .....	iii
ABSTRACT .....	iv
ÖZET .....	v
LIST OF FIGURES .....	vii
LIST OF TABLES .....	viii
LIST OF SYMBOLS/ABBREVIATIONS .....	ix
1. INTRODUCTION .....	1
1.1. Adaptive Filtering Algorithms and Trade off Problem .....	2
1.2. Methods to Remove the Trade off Problem .....	5
1.3. Scope of Thesis .....	6
2. ZERO FORCING ALGORITHMS .....	7
3. ADAPTIVE STEP SIZE AFFINE PROJECTION ALGORITHM .....	18
4. EXPONENTIALLY WEIGHTED AFFINE PROJECTION ALGORITHM .....	26
4.1. Switching Exponentially Weighted Affine Projection Algorithm .....	28
5. COMPUTER SIMULATIONS .....	30
5.1. Nonstationary System Identification .....	30
5.2. Steady State Analysis of Affine Projection Algorithm .....	39
5.3. Acoustic Echo Cancellation .....	41
5.4. Inverse System Identification .....	49
6. DISCUSSIONS ON THE PROPOSED ALGORITHMS .....	53
7. CONCLUSIONS .....	55
7.1. Suggestions for Future Work .....	55
APPENDIX A: GRADIENT-BASED ALGORITHMS .....	56
A.1. Steepest Descent Algorithm .....	56
A.2. Least Mean Square Algorithm .....	58
A.2.1. Performance Analysis of LMS algorithm .....	59
A.3. Transform Domain LMS Algorithm .....	66
REFERENCES .....	68

## LIST OF FIGURES

Figure 1.1. MSD curves for LMS and RLS algorithms to demonstrate trade off problem .....	3
Figure 2.1. System identification with filter .....	7
Figure 2.2. Transversal filter.....	8
Figure 2.3. Squared error curves for Zero Forcing algorithm and APA(M) .....	13
Figure 2.4. Mean square deviation curves for APA with different orders.....	14
Figure 2.5. Mean square deviation curves for APA(2) with different step sizes.....	15
Figure 5.1. System identification configuration .....	30
Figure 5.2. MSD curves for the APA(2) and the GASS- $\alpha$ -S-APA(2) as an example of the convergence characteristics.....	34
Figure 5.3. The step size curve for the GASS- $\alpha$ -S-APA(2) .....	34
Figure 5.4. Acoustic echo cancellation configuration .....	42
Figure 5.5. Teleconference room impulse response example.....	43
Figure 5.6. The input speech and its echo from the 256 ms echo path.....	44
Figure 5.7. Square deviation curves for the NLMS and VSS-NLMS algorithms .....	47
Figure 5.8. ERLE curves for NLMS and VSS-NLMS algorithms in 32 ms AEC .....	47
Figure 5.9. Inverse system identification configuration .....	49

## LIST OF TABLES

Table 3.1. Step size updates of the GAS-APA .....	24
Table 5.1. Simulation results for the VSS and the GAS NLMS algorithms.....	35
Table 5.2. Simulation results for the VSS and the GAS APA(2) .....	37
Table 5.3. Simulation results for the EW-APA(2), the GASS- $\alpha$ -S-EW-APA(2) and the SEW-APA(2) .....	38
Table 5.4. Results of the 32 ms AEC simulation for the NLMS, the VSS-NLMS, and the GASS- $\alpha$ -S-NLMS algorithms .....	46
Table 5.5. Results of the 256 ms AEC simulation for the subband and the fullband APA(2), the VSS-APA(2), the GASS- $\alpha$ -S-APA(2), the EW-APA(2) and the SEW-APA(2) .....	48
Table 5.6. Simulation results for the NLMS, the VSS-NLMS and the GASS- $\alpha$ -S- NLMS algorithms .....	52
Table 6.1. Computational complexities of the proposed methods.....	54

## LIST OF SYMBOLS/ABBREVIATIONS

$d$	Desired response
$\mathbf{d}$	Desired response vector
$e$	(a priori) Estimation error
$\mathbf{e}$	(a priori) Estimation error vector
$\mathbf{R}$	Autocorrelation matrix
$u$	filter input
$\mathbf{u}$	Input vector
$\mathbf{U}$	Input matrix
$\mathbf{w}$	Tap weight vector
$\delta$	Regularization parameter
$\varepsilon$	A posteriori error
$\lambda$	Eigenvalue
$\chi$	Eigenvalue spread
$\lambda$	Forgetting factor parameter
$\mu$	Step size parameter
$\rho$	Learning rate parameter
$\sigma$	Standart deviation
AEC	Acoustic Echo Cancellation
APA	Affine Projection Algorithm
AR	Autoregressive
ASS	Adaptive Step Size
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DHT	Discrete Hartley Transform
DST	Discrete Sine Transform
DWT	Discrete Wavelet Transform
ERLE	Echo Return Loss Enhancement
EW	Exponentially Weighted

FIR	Finite Impulse Response
GAMS	Gradient Adaptive Matrix Step size
GAS	Gradient Adaptive Step size
GASS	Gradient Adaptive Scalar Step size
GAFF	Gradient Adaptive Forgetting Factor
GBA	Gradient-Based Algorithms
KLТ	Karhunen-Loéve Transform
LMS	Least Mean Square
LS	Least Squares
MSD	Mean Square Deviation
MSE	Mean Square Error
NLMS	Normalized Least Mean Square
RLS	Recursive Least Squares
SDA	Steepest Descent Algorithm
TDLMS	Transform Domain Least Mean Square
VSS	Variable Step size
ZFA	Zero Forcing Algorithms

## 1. INTRODUCTION

Adaptive filtering techniques are used in a wide range of applications, including system identification, echo cancellation, adaptive equalization, adaptive noise cancellation, and adaptive beamforming. The adaptive filter is usually preferred to be Finite Impulse Response (FIR) due to the inherent stability of the structure. An adaptive filter is a self-designing filter that uses a recursive algorithm (known as adaptive filtering algorithm) to adjust the filter parameters. The algorithm starts from an initial guess, chosen based on the a priori knowledge available to the system, then refines the guess in successive iterations. The adaptive filtering algorithm is designed to reach the optimum solution. The optimum solution might be time variant, and then tracking of the optimum solution is also required.

The operation of an adaptive filter has two stages, namely convergence and steady state stages. In the convergence stage, the adaptive filter starts from an initial set of parameters and converges to the optimum solution. After the convergence stage, the adaptive filter enters the steady state stage, in which the optimum solution is tracked, if it is time variant. The most important performance criterions of an adaptive filtering algorithm are the convergence speed (or convergence rate) and the misadjustment [1]. The convergence speed describes the transient behavior of the algorithm in the convergence stage. It is usually defined as the number of iterations required for the algorithm to converge “close enough” to the optimum solution. The misadjustment describes steady state behavior of the algorithm in the steady state stage. It is a quantitative measure of the amount by which the ensemble averaged final value of the mean squared error exceeds the minimum mean-squared error produced by the optimal Wiener filter.

The adaptive filtering algorithms have different convergence speeds and misadjustment characteristics. Without having any constraint on the input process of the adaptive filter, the algorithms have increasing convergence speeds with increasing computational complexities. Moreover, an algorithm may have different convergence speeds depending on the values of its design parameters. But usually, there is a trade off between the attainable convergence speed and misadjustment values. When the algorithm parameters are adjusted to obtain faster convergence, the misadjustment becomes larger,

and vice versa. An algorithm providing both fast convergence and minimum (or near minimum) misadjustment is desirable. Our research problem is to design an algorithm, which provides both fast convergence and minimum misadjustment. Hence, this algorithm does not have the convergence speed - misadjustment trade off problem.

### **1.1. Adaptive Filtering Algorithms and Trade off Problem**

We can categorize some of the adaptive filtering algorithms into two major classes: Minimum Mean Squared Error (MSE) algorithms and Zero Forcing Algorithms (ZFA).

Minimum MSE algorithms are designed to minimize the estimation error that is defined as the difference between the desired response and the desired response estimate of the adaptive filter. Steepest Descent algorithm (SDA), its stochastic counterpart Least Mean Square (LMS) algorithm and LMS-Newton algorithm and Recursive Least Squares (RLS) algorithm are well known members of minimum MSE algorithms [1,2].

SDA, LMS algorithm and LMS-Newton algorithm are well explained with the Wiener Filter theory [1,2]. Wiener filter theory uses the minimum mean squared error criterion to obtain a large class of optimum linear filters known as Wiener Filters. In this class, the adaptive filtering algorithms that are based on the Wiener-Hopf equations and the steepest descent optimization are known as Gradient Based Algorithms (GBA). SDA, LMS algorithm and LMS-Newton algorithm are members of the GBA. In Appendix A, GBA are examined in detail.

Minimum MSE algorithms employ an averaging over the input process to get smaller misadjustment. But, this method causes a trade off between convergence speed and misadjustment. Increasing the ensemble size, smaller misadjustment can be obtained, however convergence speed becomes slower. In Figure 1.1, we illustrate the performances of the LMS and the RLS algorithms for an identification problem in a stationary environment. Mean Square Deviation (MSD) curves are plotted to demonstrate the convergence and steady state performances. At iteration 10.000, an instantaneous change occurs in the optimum solution and so algorithms converge again. We observe that when the parameters of algorithms are adjusted to obtain smaller MSD (also means smaller

misadjustment), convergence speed becomes slower. This is the convergence speed - misadjustment trade off problem.

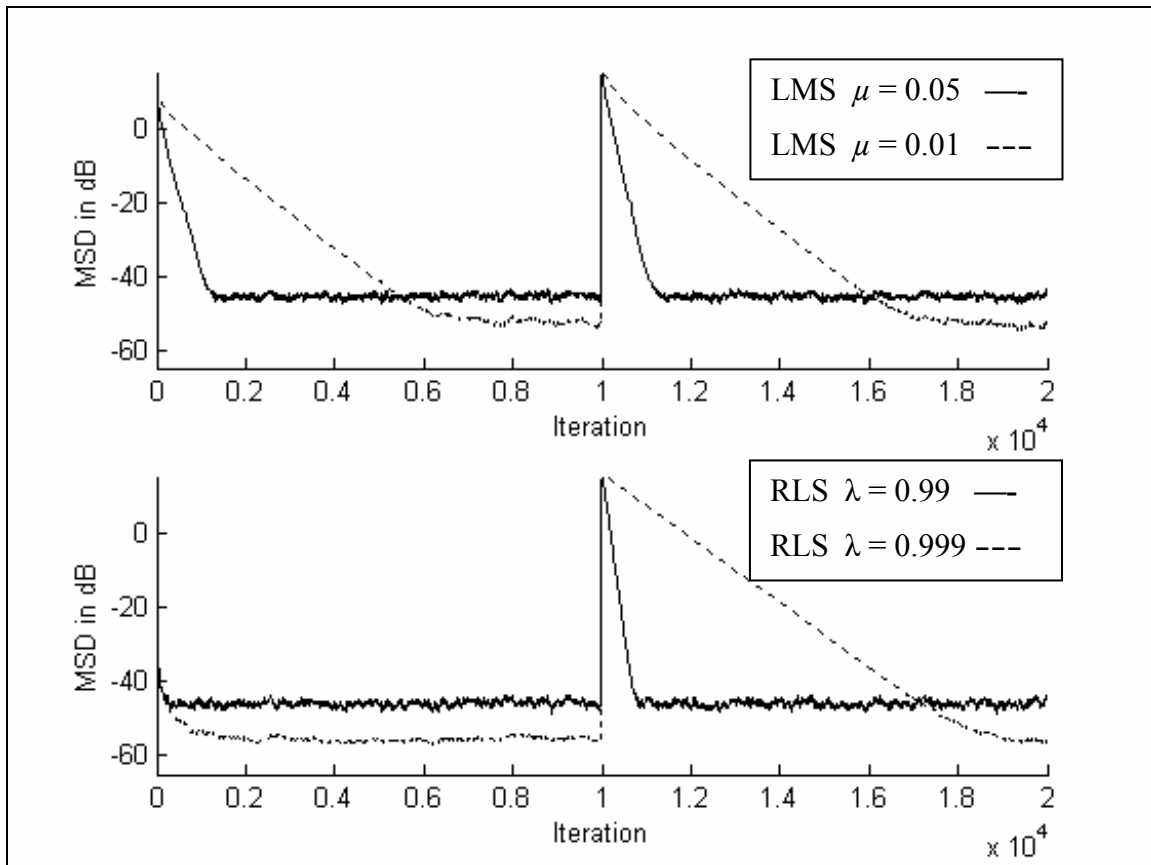


Figure 1.1. MSD curves for LMS and RLS algorithms to demonstrate trade off problem

The other class is Zero Forcing Algorithms. Even though these algorithms are well known in literature, their interpretation and classification as ZFA in the thesis is new. The class of ZFA is based on the fact that; when an FIR filter of length  $M$  is used, we can obtain the optimum solution by solving  $M$  equations with  $M$  unknowns (filter tap-weight parameters). This solution provides the a posteriori estimate of the desired response equal to the true value of the desired response, so the a posteriori estimation error is zero. This solution is said to “zero force” the a posteriori error. Algorithms zero forcing the a posteriori estimation error are called Zero Forcing Algorithms. Least Squares (LS) solution [1] is another zero forcing solution. The algorithm based on LS solution (LS algorithm) is a member of ZFA. ZFA may provide convergence of the adaptive filter in  $M$  iterations ( $M$  is the filter length) by solving the  $M \times M$  full rank equation matrix. There is another member

of ZFA, which is based on partial rank filtering equations. This algorithm is known as Affine Projection Algorithm (APA) [1]. It is also called as Partial Rank Algorithm. When the APA provides full rank solution, it becomes equivalent to the LS algorithm and may converge in  $M$  iterations. The APA provides slower convergence with the lower projection orders (partial rank solutions), and then convergence speed is also highly dependent on the correlation of the input process. The APA with projection order one is known as the Normalized LMS algorithm (NLMS) [1].

ZFA require inversion of a matrix whose elements are constituted by the input process of the filter. In the tap-weight update of the adaptive filter, ZFA use the inverse matrix and the a priori estimation error. The desired response includes a term named the measurement noise. Measurement noise is the source of the error in tap-weight estimates. When the measurement noise is zero, the optimum solution can be exactly obtained. The inverse matrix in the tap-weight update of the ZFA can have large elements. These large elements cause amplification of the measurement noise; hence the tap-weight estimate gets more erroneous. Larger errors in the tap-weight estimate cause larger (a priori) estimation error in the next iteration. This effect is known as the noise amplification of ZFA [2], and causes large variance for the a priori estimation error.

Although ZFA can provide convergence as fast as in  $M$  iterations, the computational complexity required for the  $M \times M$  matrix inversion is too cumbersome for practical application. Among ZFA, the APA that has order lower than  $M$  might provide fast convergence with reasonable computational complexity. Even with order two and highly correlated input process, the APA converges fast enough for most practical applications. Increasing the projection order, convergence speed can also be increased. But, the increase in convergence speed is not linearly dependent on the projection order and the misadjustment becomes larger. This is one form of the trade off problem with the APA and will be illustrated later in the next section where the APA is derived.

When we use a scalar (step size) multiplier less than one in the tap-weight update, the APA provides smaller misadjustment, but it also converges slower by losing the zero forcing ability. Therefore, the constant step size also introduces another form of the trade off problem, which will also be illustrated in the next section.

The facts about the minimum MSE algorithms and the ZFA bring us to this conclusion: the convergence is a problem requiring a solution which zero forces a posteriori estimation error and the steady state performance is a problem requiring a solution which minimizes a priori estimation error. Therefore, an ideal adaptive filtering algorithm provides zero forcing in convergence stages and minimizes the a priori estimation error in the steady state. To design such algorithm, one has to use a member of the ZFA and modify it to achieve both zero forcing in the convergence stage and minimum a priori estimation error in the steady state. Hence, the trade off problem of the algorithm will be removed.

## **1.2. Methods to Remove the Trade off Problem**

In literature, there are proposals to remove the trade off problem of the LMS and the RLS algorithms. Adaptive Step Size (ASS) methods for the LMS algorithm and Adaptive Forgetting Factor (AFF) methods for the RLS algorithm are proposed in that respect. However, these methods do not remove the slow convergence problem of the algorithms, which is inherited from their minimum MSE objective. These methods are aimed to conserve fast convergence that is available with a set of design parameter values and also provide smaller misadjustment by readjusting the parameter values. The ASS methods can provide an optimum step size value which is the step size value providing the minimum misadjustment. The simplest ASS method is based on the steepest descent optimization, hence called Gradient Adaptive Step size (GAS) [1-9]. The GAS-LMS algorithm converges too slowly to the optimum step size value and so to the minimum misadjustment [7]. Among the ASS methods, we call some as Variable Step Size (VSS) methods [10-15], since they are not adaptive to track the optimum step size value in a nonstationary environment. The VSS-LMS algorithms can not provide the minimum misadjustment in the tracking problem, since they can not acquire and track the optimum step size. Even they may cause worse steady state results, when the algorithm parameters are not appropriately adjusted. AFF-RLS algorithm is also based on the steepest descent optimization [1,16]. It works well only with very short filter lengths.

Our research problem is to design an algorithm providing both fast convergence and minimum (or near minimum) misadjustment. The APA is a good candidate with its fast convergence compared to that of the LMS and the RLS algorithms. But, it also has the convergence speed trade off problem, when it is adjusted to provide smaller misadjustment. In literature, there is no proposal on the APA to remove its trade off problem. In the thesis, we propose the GAS-APA to achieve both fast convergence and the minimum misadjustment. In addition, we propose the VSS-APA that has fast convergence and the minimum misadjustment, when the optimum solution is time invariant (stationary). Moreover, we propose the Exponentially Weighted APA (EW-APA) that is similar to the RLS algorithm; hence it has the trade off problem. To remove that trade off problem, we propose a switching method between the APA and the EW-APA, and call this new algorithm Switching EW-APA (SEW-APA). Hence, SEW-APA has both fast convergence of the APA and small misadjustment of the EW-APA.

### **1.3. Scope of Thesis**

In the thesis, the adaptive step size (ASS) methods for Affine Projection Algorithm (ASS-APA), the Exponentially Weighted APA (EW-APA) and an algorithm that switches between APA and EW-APA (SEW-APA) are proposed. In the second section, adaptive filtering problem and the class of Zero Forcing Algorithms are introduced. In the third section, Adaptive Step Size APA is proposed. In this section, Gradient Adaptive Step size APA (GAS-APA) and Variable Step size APA (VSS-APA) are proposed as two different types of adaptive step size. In Section 4, Exponentially Weighted APA (EW-APA) and SEW-APA are proposed. In Section 5, the proposed algorithms are compared with the computer simulation results. Nonstationary system identification, acoustic echo cancellation and nonstationary inverse system identification simulations demonstrate the performances of the algorithms. In Section 6, the algorithms are discussed and compared. Section 7 is the conclusions part of the thesis.

## 2. ZERO FORCING ALGORITHMS

Before introducing Zero Forcing Algorithms (ZFA), we have to understand a basic solution for the filtering problem. System identification model in Figure 2.1 will be the basic system to illustrate the filtering fundamentals.

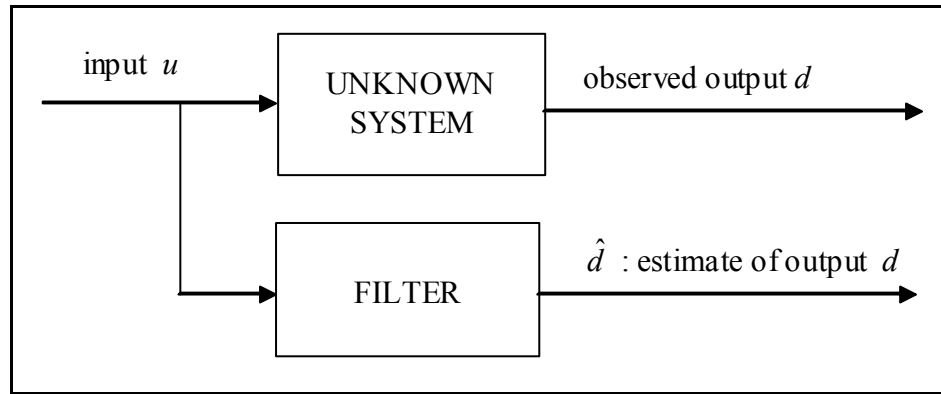


Figure 2.1. System identification with filter

Firstly, we will define a trivial solution for the system identification: If we give an input of single impulse into the unknown system, the observed sequence  $d(n)$  (known as desired response) constitutes the impulse response of the unknown system, which is also equal to the optimum solution (optimum solution tap-weight vector:  $\mathbf{w}_{opt}$ ). The single impulse must be isolated such that the input samples before and after the impulse must be zero. When we use a transversal filter of order  $M$  with tap-weight vector  $\mathbf{w}(n)$  at time  $n$  in Figure 2.1 and assume that the optimum solution  $\mathbf{w}_{opt}$  is also of order  $M$ , we can obtain the optimum solution in  $M$  iterations by using the single impulse. The single impulse input  $u(n)$  to the filter and the unknown system is as:

$$\begin{aligned} u(n) &= a & n &= 1 \\ u(n) &= 0 & n &= 2, 3, \dots, M \text{ and } n < 1 \end{aligned} \quad (2.1)$$

where  $a$  is the constant amplitude of the delta dirac function. In Figure 2.2, the transversal filter structure is illustrated.

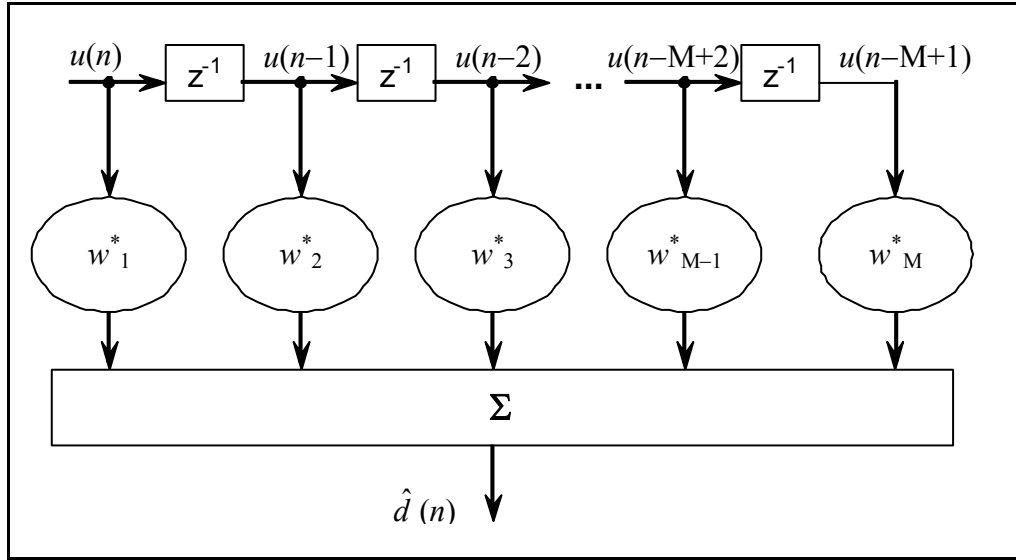


Figure 2.2. Transversal filter

To obtain the estimate of the desired response and the optimum tap-weight solution, each tap-weight of the transversal filter in Figure 2.2 can be calculated as:

$$w_k(n) = \frac{d^*(n)}{u^*(n-k)} \quad n = 1, 2, \dots, M \quad (2.2)$$

where  $w_k(n)$  is the  $k^{\text{th}}$  tap-weight of the filter and it is updated only when  $n$  is equal to  $k$ ,  $u(n-k)$  is the input sample to  $k^{\text{th}}$  tap-weight and it is equal to  $a$  from  $n = 1$  to  $n = M$ . Hence, the update equation (2.2) can be simplified as:

$$w_k(n) = \frac{d^*(n)}{a} \quad n = 1, 2, \dots, M \text{ and } k = n \quad (2.3)$$

Theoretically, we do not need the isolated single impulse input process defined in (2.1) to get the optimum solution in  $M$  iterations. Having an input process that is led by  $M-1$  zeros; we could also update the tap-weights of the filter such that the optimum solution could be obtained in  $M$  iterations. This update procedure is defined below:

$$e(n) = d(n) - \mathbf{w}^H(n)\mathbf{u}(n) \quad (2.4)$$

$$w_k(n) = \frac{e^*(n)}{u^*(n-k)} \quad (2.5)$$

for  $n = 1, 2, 3, \dots, M$

where  $e(n)$  is the (a priori) estimation error,  $w_k(n)$  is updated only when  $n$  is equal to  $k$  and  $\varepsilon(n)$  is the a posteriori error and equals zero along the filtering process:

$$\varepsilon(n) = d(n) - \mathbf{w}^H(n+1)\mathbf{u}(n) = 0 \quad (2.6)$$

The algorithm defined in (2.4) and (2.5) is executed only for first  $M$  iterations and it can give convergence in  $M$  iterations only when  $u(n)$  is zero for  $n < 1$  (leading zeros).

We can also write a new algorithm, which can converge in  $M$  iterations without having the leading zeros constraint over the input process. Additionally, this algorithm can track the changes in optimum solution by operating for iterations greater than  $M$ . Before stating the algorithm, we write the filtering equation on which the algorithm is based on:

$$\mathbf{U}(n)\mathbf{w}_{opt}^*(n) = \mathbf{d}(n) \quad (2.7)$$

where  $\mathbf{w}_{opt}(n)$  is the optimum solution which is time variant (nonstationary),  $\mathbf{U}(n)$  is the  $M \times M$  input matrix ( $M$ : length of  $\mathbf{w}_{opt}$ ),  $\mathbf{U}(n)$  is equal to  $\mathbf{U}^T(n)$  and  $\mathbf{d}(n)$  is the desired response vector:

$$\mathbf{U}(n) = [ \mathbf{u}(n) \ \mathbf{u}(n-1) \ \mathbf{u}(n-2) \ \dots \ \mathbf{u}(n-M+1) ] \quad (2.8)$$

where

$$\mathbf{u}(n) = [ u(n) \ u(n-1) \ \dots \ u(n-M+1) ]^T \quad (2.9)$$

$$\mathbf{d}(n) = [ d(n) \ d(n-1) \ \dots \ d(n-M+1) ]^T \quad (2.10)$$

From (2.7), it is clear that optimum solution can be calculated as follows.

$$\mathbf{w}_{opt}^*(n) = \mathbf{U}^{-1}(n)\mathbf{d}(n) \quad (2.11)$$

To be able to obtain the optimum solution in the  $M^{\text{th}}$  iteration from (2.11), the input matrix  $\mathbf{U}(n)$  must be invertible.

The algorithm based on filtering equation (2.7) and batch solution (2.11) can be stated as follows:

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{U}(n)\mathbf{w}^*(n) \quad (2.12)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + [\mathbf{U}^*(n)]^{-1}\mathbf{e}^*(n) \quad (2.13)$$

This algorithm will have zero a posteriori error as follows:

$$\boldsymbol{\varepsilon}(n) = \mathbf{d}(n) - \mathbf{U}(n)\mathbf{w}^*(n+1) = \mathbf{0} \quad (2.14)$$

and a priori error vector will have only one non-zero element in the first entry when the input matrix  $\mathbf{U}(n)$  is invertible:

$$\mathbf{e}(n) = [e(n) \ 0 \ 0 \ \dots \ 0]^T \quad (2.15)$$

We name the algorithm defined in Equations (2.12) and (2.13) as the Zero Forcing Algorithm. The name comes from the zero a posteriori error property of the algorithm.

We could write another form of the filtering equation (2.7) by multiplying both sides with  $\mathbf{U}^H(n)$ , taking complex conjugate and using the fact that  $\mathbf{U}^T(n) = \mathbf{U}(n)$ :

$$\mathbf{U}(n)\mathbf{U}^*(n)\mathbf{w}_{opt}(n) = \mathbf{U}(n)\mathbf{d}^*(n) \quad (2.16)$$

The form in (2.16) is known as Least Squares (LS) solution [1,2] and an algorithm (LS algorithm) could also be written based on the LS solution as:

$$\mathbf{e}(n) = \mathbf{d}(n) - \mathbf{U}(n)\mathbf{w}^*(n) \quad (2.17)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + [\mathbf{U}(n)\mathbf{U}^*(n)]^{-1}\mathbf{U}(n)\mathbf{e}^*(n) \quad (2.18)$$

This algorithm will also have zero a posteriori error as follows:

$$\boldsymbol{\varepsilon}(n) = \mathbf{d}(n) - \mathbf{U}(n)\mathbf{w}^*(n+1) = \mathbf{0} \quad (2.19)$$

and a priori error vector will have only one non-zero element in the first entry when the matrix  $\mathbf{U}(n)\mathbf{U}^*(n)$  is invertible:

$$\mathbf{e}(n) = [e(n) \ 0 \ 0 \ \dots \ 0]^T \quad (2.20)$$

Using the property defined in (2.20), the tap-weight update (2.18) can be rewritten in another form as follows:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + [\mathbf{U}(n)\mathbf{U}^*(n)]^{-1}\mathbf{u}(n)e^*(n) \quad (2.21)$$

The form in (2.21) is equivalent to the LS algorithm only when the matrix  $\mathbf{U}(n)\mathbf{U}^*(n)$  is invertible. When it is not invertible and so regularization is applied to get an invertible matrix, this form converges slower than the LS algorithm. The Recursive Least Squares algorithm can be shown to be based on the form in (2.21), where the inverse matrix estimate is replaced with the inverse of the exponentially weighted, time averaged correlation matrix. The averaging operation removes the effect of measurement noise, hence it provides smaller misadjustment.

Until now, we have introduced the algorithms, which involve estimation of the inverse of an  $M \times M$  matrix. These algorithms provide the optimum solution in  $M$  iterations, when the matrix to be inverted is a full rank matrix. However, there is a computationally simpler algorithm, which can provide convergence in  $M$  iterations by employing a special constraint on input process  $u(n)$  which will be explained in subsequent paragraphs. In this algorithm, the reduction of computational complexity comes mainly from the inverse matrix estimation process. The inverse matrix estimate is done for a  $P \times P$  matrix,  $P < M$ . When we remove the special constraint on input process, this algorithm loses the ability to converge in  $M$  iterations. However, the algorithm exhibits increasing convergence speeds with the increasing order  $P$ . The algorithm becomes equivalent to the LS algorithm when  $P$  equals  $M$ , hence it can converge in  $M$  iterations. This algorithm is known as the Affine

Projection Algorithm or the Partial Rank Algorithm. In the following, we will derive the APA.

We can rewrite the filtering equation (2.7) with a partial rank input matrix as follows:

$$\mathbf{U}_p^T(n)\mathbf{w}_{opt}^*(n) = \mathbf{d}_p(n) \quad (2.22)$$

where  $\mathbf{U}_p(n)$  is the  $M \times P$  input matrix and  $\mathbf{d}_p(n)$  is  $P \times 1$  desired response vector:

$$\mathbf{U}_p(n) = [ \mathbf{u}(n) \ \mathbf{u}(n-1) \ \mathbf{u}(n-2) \ \dots \ \mathbf{u}(n-P+1) ] \quad (2.23)$$

$$\mathbf{d}_p(n) = [ d(n) \ d(n-1) \ d(n-2) \ \dots \ d(n-P+1) ]^T \quad (2.24)$$

where  $\mathbf{u}(n)$  is defined by (2.9). Partial rank filtering equation (2.22) could be rewritten as:

$$\mathbf{U}_p [ \mathbf{U}_p^H \mathbf{U}_p ]^{-1} \mathbf{U}_p^H \mathbf{w}_{opt} = \mathbf{U}_p [ \mathbf{U}_p^H \mathbf{U}_p ]^{-1} \mathbf{d}_p^* \quad (2.25)$$

where time index  $n$  is dropped for the clarity and  $\mathbf{U}_p [ \mathbf{U}_p^H \mathbf{U}_p ]^{-1} \mathbf{U}_p^H$  is known as the projection operator in the LS literature [1].

Now, we can write the APA based on the partial rank filtering equation:

$$\mathbf{e}_p(n) = \mathbf{d}_p(n) - \mathbf{U}_p^T(n)\mathbf{w}^*(n) \quad (2.26)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mathbf{U}_p(n)[\mathbf{U}_p^H(n)\mathbf{U}_p(n)]^{-1}\mathbf{e}_p^*(n) \quad (2.27)$$

APA will also have zero a posteriori error  $\boldsymbol{\varepsilon}_p(n)$  as follows:

$$\boldsymbol{\varepsilon}_p(n) = \mathbf{d}_p(n) - \mathbf{U}_p^T(n)\mathbf{w}^*(n+1) = \mathbf{0} \quad (2.28)$$

and the a priori error vector  $\mathbf{e}_p(n)$  will have only one non-zero element in the first entry:

$$\mathbf{e}_p(n) = [ e(n) \ 0 \ 0 \ \dots \ 0 ]^T \quad (2.29)$$

The algorithms defined above forces the a posteriori error to zero. Therefore, they are classified under the name Zero Forcing Algorithms. ZFA having the full rank solution converges in  $M$  iterations, without need to the leading zeros constraint on the input. ZFA converging in first  $M$  iterations have the same tap-weight update of the algorithm defined in (2.4) and (2.5). That is to say, the tap-weight update term has only one non-zero element at the  $n^{\text{th}}$  row from  $n = 1$  to  $M$ .

In Figure 2.3, the squared error curves for the Zero Forcing algorithm (defined in Equations (2.12) and (2.13)) and the APA(M) (having projection order  $M$ , and is equivalent to the LS algorithm) are plotted for a stationary system identification problem. The curves demonstrate the convergence in approximately  $M$  iterations (not exactly  $M$ , because of the regularization used in the matrix inversion operations). In the steady state, we observe large error values due to the noise amplification.

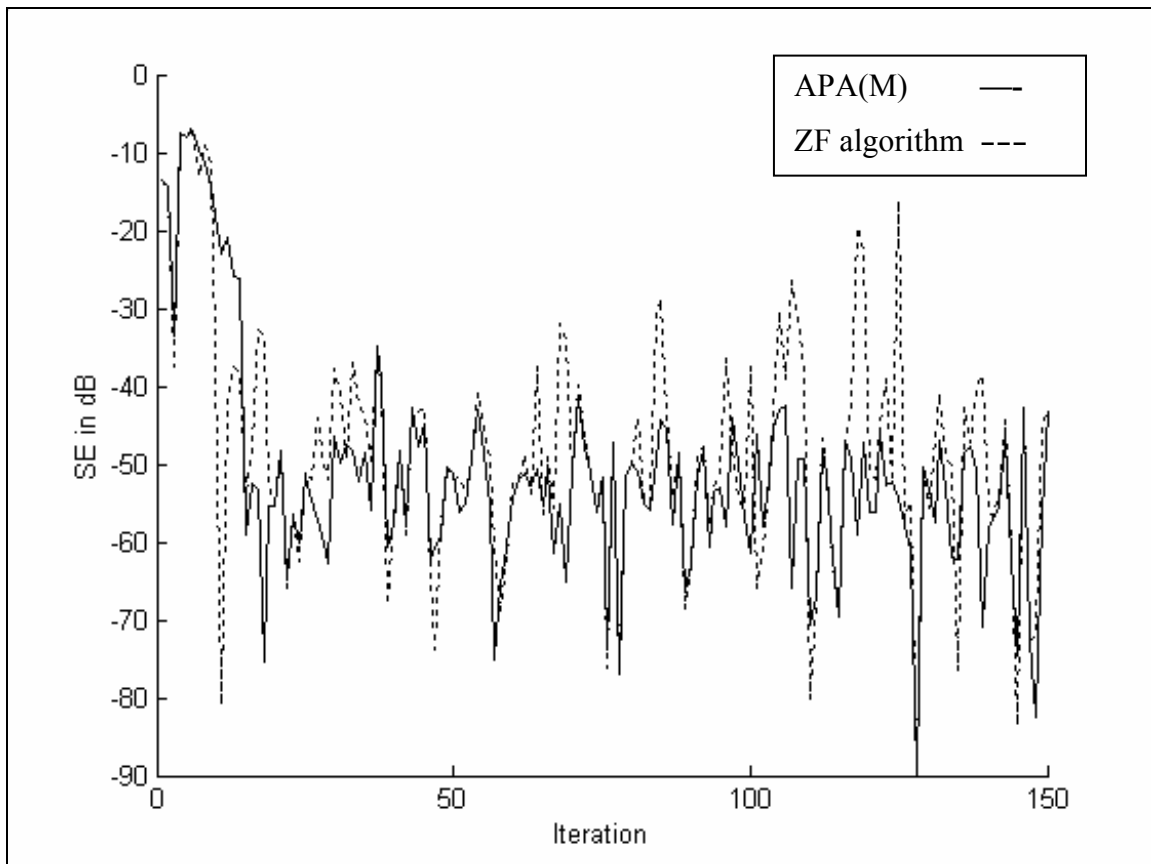


Figure 2.3. Squared error curves for Zero Forcing algorithm and APA(M)

As we stated previously, the APA has a projection order dependent convergence characteristic and there is also a convergence speed – misadjustment trade off problem with changing projection order. In Figure 2.4, the order dependent trade off problem is demonstrated by plotting the mean square deviation curves for different projection orders.

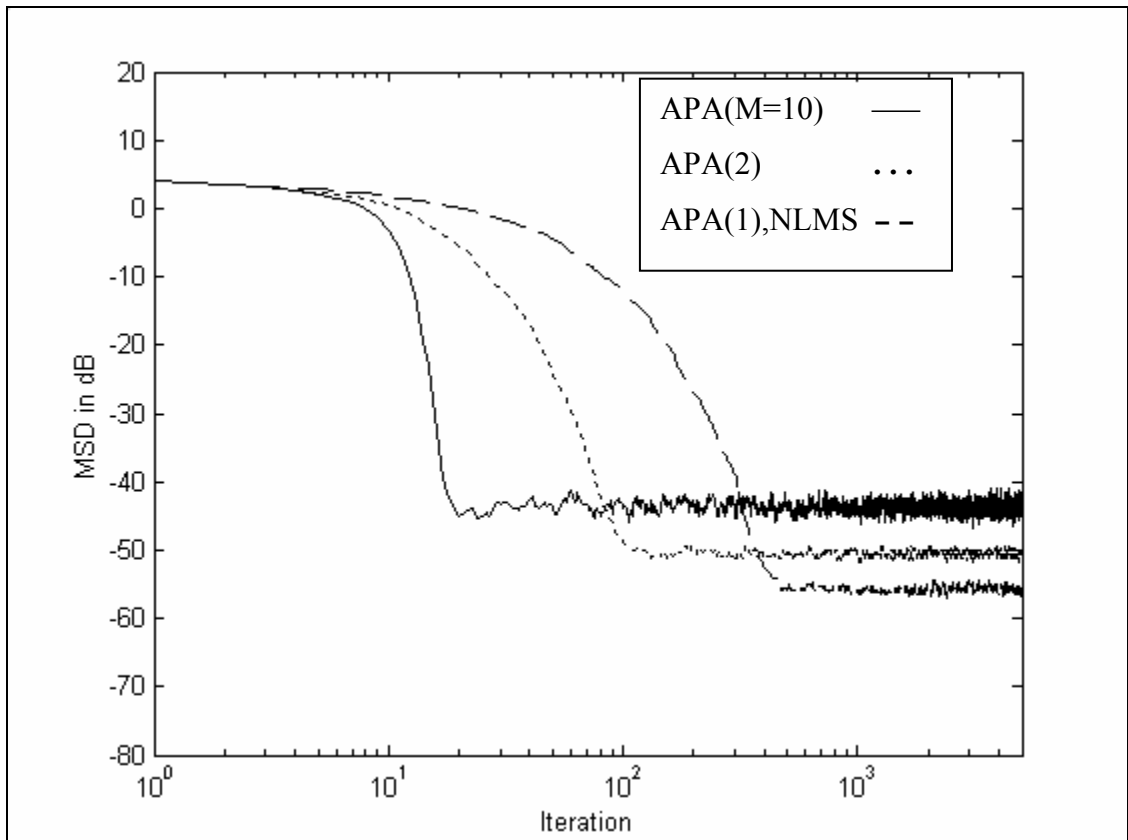


Figure 2.4. Mean square deviation curves for APA with different orders

The trade off problem emerges in another way, when we use a scalar step size in the tap-weight update of the APA. In Figure 2.5, this form of the trade off problem is demonstrated.

The APA with order  $P < M$  requires a special constraint on the input process  $u(n)$  (additional to the leading zeros constraint) to achieve convergence in the first  $M$  iterations. This special constraint that  $u(n)$  must satisfy can be stated as:

$$\mathbf{U}_p(n) [\mathbf{U}_p^H(n) \mathbf{U}_p(n)]^{-1} = \mathbf{B}(n) \quad \text{for } n = 1 \text{ to } M \quad (2.30)$$

where  $\mathbf{B}(n)$  is a matrix and its first column vector has its  $n^{\text{th}}$  row element equal to the inverse of  $u(1)$  and other elements equal to zero. Some  $u(n)$  sequences satisfying the constraint in (2.30) are given in (2.31):

$$\begin{aligned}\mathbf{u}(n) &= [a \ a \ a \ a \ \dots \ a]^T \\ \mathbf{u}(n) &= [a \ -a \ a \ -a \ a \ \dots \ a]^T\end{aligned}\quad (2.31)$$

where  $a$  is a scalar. These two sequences can provide convergence in first  $M$  iterations when  $1 < P < M$  in the APA. The APA that has  $P = 1$  (NLMS algorithm or Projection Algorithm) can provide convergence in first  $M$  iterations, only when the single impulse training sequence is used.

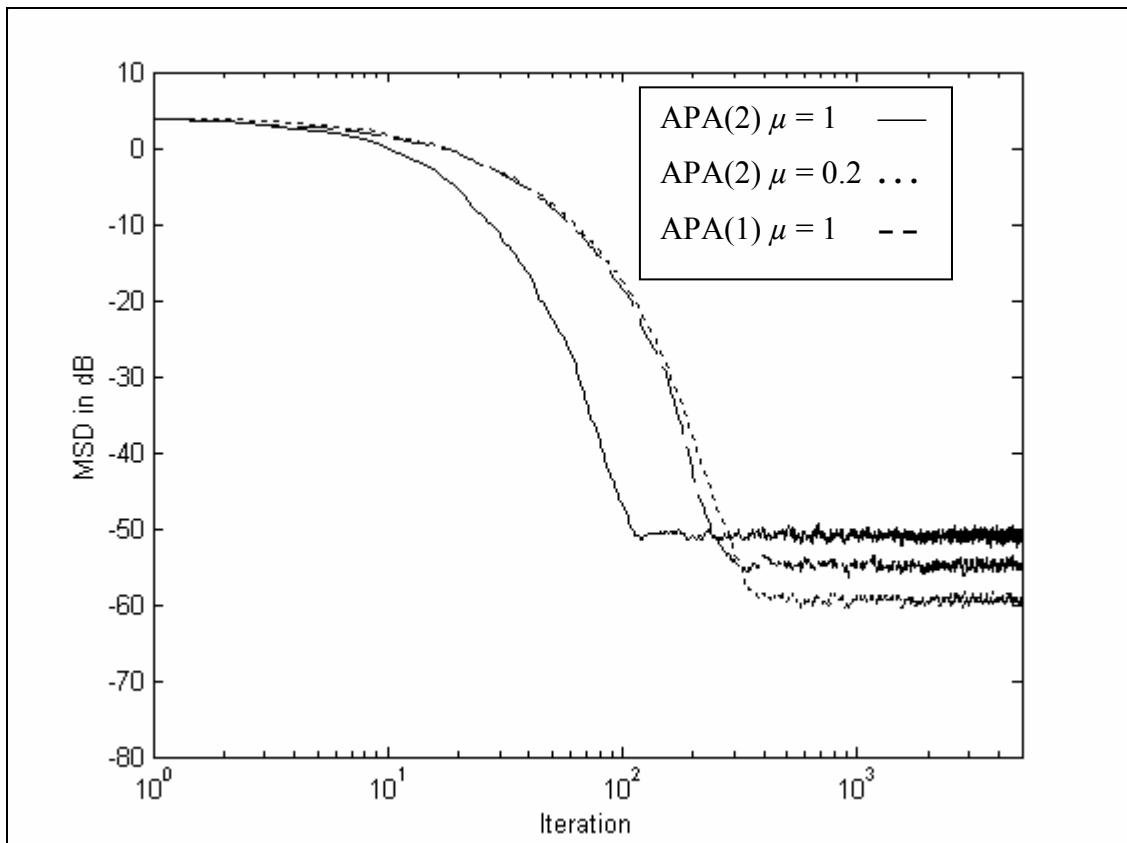


Figure 2.5. Mean square deviation curves for APA(2) with different step sizes

Now, we will explain how zero forcing brings fast convergence to the ZFA. Firstly, we write the adaptive filtering equations:

$$d(n) = \mathbf{w}_{opt}^H(n)\mathbf{u}(n) \quad (2.32)$$

$$\hat{d}(n) = \mathbf{w}^H(n)\mathbf{u}(n) \quad (2.33)$$

$$e(n) = d(n) - \hat{d}(n) \quad (2.34)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \Delta\mathbf{w}(n) \quad (2.35)$$

$$\varepsilon(n) = d(n) - \mathbf{w}^H(n+1)\mathbf{u}(n) \quad (2.36)$$

$$e(n+1) = d(n+1) - \mathbf{w}^H(n+1)\mathbf{u}(n+1) \quad (2.37)$$

where  $d(n)$  is the desired response (measurement noise ignored),  $\mathbf{w}_{opt}(n)$  is the optimum solution,  $\mathbf{u}(n)$  is the input vector to adaptive filter,  $\hat{d}(n)$  is the estimate of the desired response,  $\mathbf{w}(n)$  is the tap-weight vector of the adaptive filter,  $e(n)$  is the a priori estimation error,  $\varepsilon(n)$  is the a posteriori estimation error,  $\Delta\mathbf{w}(n)$  is the tap-weight update vector all at time  $n$ .

By zero forcing the a posteriori error,  $\mathbf{w}(n+1)$  can be equal to  $\mathbf{w}_{opt}(n)$  at iteration  $M$  (with full rank solution) and the a priori estimation error is zero at the  $(M+1)^{\text{th}}$  iteration and later.

Now, we could make our model more realistic. In real life, the optimum solution may be time variant (nonstationary system). Although  $\mathbf{w}(n+1)$  can be equal to  $\mathbf{w}_{opt}(n)$  by zero forcing, the a priori estimation error never becomes zero due to variations in the optimum solution ( $\mathbf{w}_{opt}(n+1)$  is not equal to  $\mathbf{w}_{opt}(n)$ ). ZFA may converge in  $M$  iterations (squared (a priori) estimation error converges at the  $(M+1)^{\text{th}}$  iteration).

Even when the optimum solution is stationary, the a priori estimation error can not reach to zero, because the desired response  $d(n)$  includes a term known as the measurement noise. The desired response (2.32) must be restated including the measurement noise  $v(n)$ :

$$d(n) = \mathbf{w}_{opt}^H(n)\mathbf{u}(n) + v(n) \quad (2.38)$$

The measurement noise and the time variation set the performance bound on the estimates of the desired response and so the estimation error. But, they do not prevent convergence of the ZFA in  $M$  iterations.

The main drawback of ZFA is that: they give large values of misadjustment. The estimation error includes an unwanted noise term (measurement noise). The noisy estimation error and the input process are used to update the tap-weights. Most ZFA have an inverse matrix estimate based on the input process in the tap-weight update. The inverse matrix could have very large values. The elements with large values amplify the disturbing effect of the measurement noise on the tap-weight estimate. A large deviation in the estimated tap-weight vector causes higher estimation error in the next iteration.

Although zero forcing gives fast convergence, associated misadjustment is large due to the noise amplification effect of zero forcing. To obtain better misadjustment, one method is to use the estimate of the inverse autocorrelation matrix instead of the estimates of the inverse matrices (in Equations (2.18) and (2.27)) in the ZFA. Using estimate of the inverse autocorrelation matrix on the other hand causes slower convergence, while it provides smaller misadjustment. In the thesis, we will propose the use of this method with the APA, naming the new algorithm the Exponentially Weighted APA.

Another method is to use a multiplier term (step size) in the tap-weight update term as in (2.39). When the step size is also optimized to give minimum estimation error, it is called Adaptive Step Size (ASS).

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu(n)\Delta\mathbf{w}(n) \quad (2.39)$$

where  $\mu(n)$  is the adaptive step size (scalar or diagonal matrix) optimized to provide minimum estimation error in steady state, and it is upper bounded to one to provide zero forcing during convergence. In the next section, we will propose ASS-APA as an algorithm providing both fast convergence and minimum misadjustment.

### 3. ADAPTIVE STEP SIZE AFFINE PROJECTION ALGORITHM

The APA is known to have increasing convergence speeds with the increasing order  $P$  [17]. When the order  $P$  equals the filter length  $M$ , convergence in nearly  $M$  iterations is possible. It may not be exactly  $M$  iterations due to the regularization which is employed to avoid the inversion of a rank deficient matrix. The convergence speed improvement with the increasing order  $P$  is not linear. The increased computational cost with increasing order  $P$  is expensive regarding the gained convergence speed improvement. In addition, increasing the order  $P$  to improve convergence also leads to worse steady state performance. In the thesis, we use an adaptive step size parameter in the tap-weight update of the APA, hence we obtain an algorithm providing both fast convergence and minimum misadjustment.

In literature, there are adaptive step size methods proposed for the LMS algorithm [1-9], which provide the optimum step size to obtain minimum MSE. These adaptive step size methods use steepest descent optimization and MSE criteria, hence relatively low computational complexity is required to implement. They are also known as Gradient Adaptive Step size methods. In addition, different variable step size methods [10-15] are also introduced to overcome convergence speed-misadjustment trade off problem. The VSS method is especially useful when the optimum solution is time invariant.

In the thesis, we propose gradient adaptive step size method for APA. The GAS-APA provides both fast convergence associated with zero forcing and minimum misadjustment associated with the optimum step size. In the following, we derive the GAS-APA.

Firstly, we restate the APA update equations with an adaptive step size parameter:

$$\mathbf{e}_p(n) = \mathbf{d}_p(n) - \mathbf{U}_p^T(n)\mathbf{w}^*(n) \quad (3.1)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu(n)\mathbf{U}_p(n)\hat{\Phi}^{-1}(n)\mathbf{e}_p^*(n) \quad (3.2)$$

$$\Phi(n) = \mathbf{U}_p^H(n)\mathbf{U}_p(n) \quad (3.3)$$

Below, we define the MSE cost function:

$$J(n) = E[e(n)e^*(n)] \quad (3.4)$$

where

$$e(n) = d(n) - \mathbf{u}^T(n)\mathbf{w}^*(n) \quad (3.5)$$

Differentiating the cost function  $J(n)$  with respect to the step size parameter  $\mu(n)$  yields the scalar gradient:

$$\nabla_{\mu}(n) = \frac{\partial J(n)}{\partial \mu(n)} \quad (3.6)$$

$$\nabla_{\mu}(n) = E \left[ \frac{\partial e(n)}{\partial \mu(n)} e^*(n) + \frac{\partial e^*(n)}{\partial \mu(n)} e(n) \right] \quad (3.7)$$

From (3.5), we obtain

$$\frac{\partial e(n)}{\partial \mu(n)} = -\mathbf{\Psi}^H(n)\mathbf{u}(n) \quad (3.8)$$

where  $\mathbf{\Psi}(n)$  denotes the gradient of the tap-weight vector  $\mathbf{w}(n)$  with respect to the step size parameter:

$$\mathbf{\Psi}(n) = \frac{\partial \mathbf{w}(n)}{\partial \mu(n)} \quad (3.9)$$

Now, we can redefine the scalar gradient as

$$\nabla_{\mu}(n) = -E \left[ \mathbf{\Psi}^H(n)\mathbf{u}(n)e^*(n) + \mathbf{u}^H(n)\mathbf{\Psi}(n)e(n) \right] \quad (3.10)$$

The gradient adaptive step size update equation can be written as

$$\mu(n+1) = \mu(n) - \frac{1}{2} \rho \hat{\nabla}_{\mu}(n) \quad (3.11)$$

where  $\rho$  is a small, positive learning rate parameter and  $\hat{\nabla}_{\mu}(n)$  is an estimate of the scalar gradient.

In the GAS update equation, we could employ the instantaneous estimate of the scalar gradient in (3.10):

$$\hat{\nabla}_{\mu}(n) = -2 \operatorname{Re} \left[ \hat{\Psi}^H(n) \mathbf{u}(n) e^*(n) \right] \quad (3.12)$$

Differentiating (3.2) with respect to the step size parameter, we have

$$\begin{aligned} \hat{\Psi}(n+1) &= \hat{\Psi}(n) + \mathbf{g}(n) + \mu(n) \mathbf{H}(n) \frac{\partial \mathbf{e}_p^*(n)}{\partial \mu(n)} \\ \mathbf{H}(n) &= \mathbf{U}_p(n) \hat{\Phi}^{-1}(n) \\ \mathbf{g}(n) &= \mathbf{H}(n) \mathbf{e}_p^*(n) \end{aligned} \quad (3.13)$$

where  $\mathbf{g}(n)$  is the projected gradient vector.

We could write the derivative of the error vector as

$$\frac{\partial \mathbf{e}_p^*(n)}{\partial \mu(n)} = [-\mathbf{u}^H(n) \hat{\Psi}(n) \ 0 \ 0 \ \dots \ 0]^T \quad (3.14)$$

Placing (3.14) in (3.13), we could write

$$\hat{\Psi}(n+1) = \hat{\Psi}(n) - \mu(n) \mathbf{h}(n) \mathbf{u}^H(n) \hat{\Psi}(n) + \mathbf{g}(n) \quad (3.15)$$

where  $\mathbf{h}(n)$  is the first column vector of the matrix  $\mathbf{H}(n)$ . Equation (3.15) can be rearranged as:

$$\hat{\Psi}(n+1) = [\mathbf{I} - \mu(n) \mathbf{h}(n) \mathbf{u}^H(n)] \hat{\Psi}(n) + \mathbf{g}(n) \quad (3.16)$$

The form in (3.16) will be used for further simplification of the GAS in the following.

Now, the GAS-APA could be summarized as follows:

Starting with initial values,  $\mathbf{w}(0)$ ,  $\mu(0)$  and  $\hat{\Psi}(0)$

$$\mathbf{e}_p(n) = \mathbf{d}_p(n) - \mathbf{U}_p^T(n)\mathbf{w}^*(n) \quad (3.17)$$

$$\Phi(n) = \mathbf{U}_p^H(n)\mathbf{U}_p(n) \quad (3.18)$$

$$\mathbf{H}(n) = \mathbf{U}_p(n)\hat{\Phi}^{-1}(n) \quad (3.19)$$

$$\mathbf{g}(n) = \mathbf{H}(n)\mathbf{e}_p^*(n) \quad (3.20)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu(n)\mathbf{g}(n) \quad (3.21)$$

$$\mu(n+1) = \left[ \mu(n) + \rho \operatorname{Re} \left[ \hat{\Psi}^H(n)\mathbf{u}(n)e^*(n) \right] \right]_{\mu^-}^{\mu^+} \quad (3.22)$$

$$\hat{\Psi}(n+1) = \hat{\Psi}(n) - \mu(n)\mathbf{h}(n)\mathbf{u}^H(n)\hat{\Psi}(n) + \mathbf{g}(n) \quad (3.23)$$

where vector  $\mathbf{h}(n)$  is the first column of the matrix  $\mathbf{H}(n)$ . Initial values  $\mathbf{w}(0)$  and  $\hat{\Psi}(0)$  are set to zero when no a priori information about them is available.  $\mu(0)$  is set to the value that provides fastest convergence speed for the initial convergence stage.

In (3.22), the bracket followed by  $\mu^-$  and  $\mu^+$  indicates truncation. The step size update must have an upper bound ( $\mu^+$  or  $\mu_{\max}$ ) considering both stability and fastest convergence. The APA gives fastest convergence with its zero forcing step size value of one and it is stable for values less than two. Step size values greater than one do not improve the convergence speed of the ZFA. The lower bound on the step size ( $\mu^-$  or  $\mu_{\min}$ ) has relatively insignificant role and it may be set to zero or some small positive number.

Instead of using only one scalar step size, M different scalar step sizes  $\mu_i$  could also be employed in a diagonal matrix step size. Then, the step size update equations (3.21), (3.22) and (3.23) become as follows:

$$\mathbf{w}_i(n+1) = \mathbf{w}_i(n) + \mu_i(n)\mathbf{g}_i(n) \quad (3.24)$$

$$\mu_i(n+1) = \left[ \mu_i(n) + \rho_i \operatorname{Re} \left[ \hat{\Psi}_i^*(n) \mathbf{u}_i(n) e^*(n) \right] \right]_{\mu^-}^{\mu^+} \quad (3.25)$$

$$\hat{\Psi}_i(n+1) = [1 - \mu_i(n) \mathbf{h}_i(n) \mathbf{u}_i^*(n)] \hat{\Psi}_i(n) + \mathbf{g}_i(n) \quad (3.26)$$

where subscript  $i$  mean  $i^{\text{th}}$  element of the vector and  $\mu_i$  is the  $i^{\text{th}}$  diagonal element of the step size matrix. In the thesis, the two forms of the step size are named the Gradient Adaptive Scalar Step size (GASS) and the Gradient Adaptive Matrix Step size (GAMS). The matrix adaptive step size may give better steady state performance than the scalar counterpart, because it assigns a unique adaptive step size to every tap-weight.

In (3.22), the step size is updated by adding the update term. Its multiplicative form is proposed in [6]. The multiplicative GASS for APA can be written as follows:

$$\mu(n+1) = \left[ \mu(n) \left( 1 + \rho \operatorname{Re} \left[ \hat{\Psi}^H(n) \mathbf{u}(n) e^*(n) \right] \right) \right]_{\mu^-}^{\mu^+} \quad (3.27)$$

We observed that the multiplicative update form shows a bad step size update behavior (leading to slow convergence), when an instantaneous change occurs in the optimum solution. In the proposal [6], behavior of the multiplicative update method for the instantaneous change is not demonstrated.

In [5], a further simplification to the GAS is proposed to decrease the computational complexity without sacrificing convergence and steady state performances. We could also employ this simplification for the GAS-APA. The simplification introduces a scalar multiplier into (3.16) instead of the matrix in the first addition term:

$$\hat{\Psi}(n+1) = \alpha \hat{\Psi}(n) + \mathbf{g}(n) \quad (3.28)$$

where  $\alpha$  is a positive constant close to and less than one. In the thesis, this form is stated with a “ $\alpha$ ” remark following the GAS name (GASS- $\alpha$ ). In [7], a simpler form of GASS- $\alpha$  ( $\alpha = 0$ ) is proposed. Having  $\alpha$  equal zero,  $\hat{\Psi}(n+1)$  is equal to projected gradient  $\mathbf{g}(n)$ . In this form, the scalar stochastic gradient for the step size update becomes less accurate and

exhibits large fluctuations. To prevent such fluctuations, relatively smaller learning rate parameter  $\rho$  is required. But then, the algorithm becomes less sensitive to the instantaneous changes of the optimum solution and the step size (and so the tap-weight estimate and the MSE curve) converges too slowly to the optimum value. This slow convergence behavior makes the GAS in [7] unattractive. To overcome this problem, a modification is proposed in [8], but this modification is totally destructive to the gradient adaptation of the step size and blindly suppresses the step size to smaller values (similar to Variable Step size (VSS) methods, we will discuss later). In [9], a different simplification is applied to the matrix version of the GAS in [7], GAMS- $\alpha$  ( $\alpha = 0$ ). In this method, the successive sign changes are used to update the matrix step size. But, this simplification leads more fluctuating step size and so worse tracking performance than GASS- $\alpha$  ( $\alpha = 0$ ).

In the thesis, we also propose a further simplification to the GAS. In the step size update (3.22), we can introduce the use of the signum function as follows:

$$\mu(n+1) = \left[ \mu(n) + \rho \operatorname{Re} \left[ \hat{\Psi}^H(n) \operatorname{sign}(\mathbf{u}(n)) e^*(n) \right] \right]_{\mu^-}^{\mu^+} \quad (3.29)$$

Using signum function ( $\operatorname{sign}(\cdot)$ ), the vector dot product between  $\hat{\Psi}^H(n)$  and  $\mathbf{u}(n)$  is simplified to sign bit multiplications and sum of elements of the resulting vector. This simpler step size update could be combined with the simplified GAS- $\alpha$ . The resulting simpler GAS is named the Gradient Adaptive Scalar Step size  $\alpha$  with Signum, GASS- $\alpha$ -S. Its matrix step size form is also possible: Matrix Step size  $\alpha$  with Signum (GAMS- $\alpha$ -S).

In Table 3.1, Gradient Adaptive Step size Affine Projection Algorithms are summarized with the step size update equations.

In literature [10-15], there are step size variation methods, which are not adaptive methods. Hence, here they are named as Variable Step size methods. These methods are not optimized to search and track for the optimum step size value. They are based on the fact that: as the algorithm proceeds from the convergence to the steady state, the estimation error gets smaller. When we design a statistic, which indicates the convergence and the

steady state, we can use this statistic to update the step size such that smaller step size is used in the steady state than that of the convergence.

Table 3.1. Step size updates of the GAS-APA

Algorithm Name	Step size update equations
Gradient Adaptive Scalar Step size, GASS	$\mu(n+1) = \left[ \mu(n) + \rho \operatorname{Re} \left[ \hat{\Psi}^H(n) \mathbf{u}(n) e^*(n) \right] \right]_{\mu-}^{\mu+}$ $\hat{\Psi}(n+1) = \hat{\Psi}(n) - \mu(n) \mathbf{h}(n) \mathbf{u}^H(n) \hat{\Psi}(n) + \mathbf{g}(n)$
Gradient Adaptive Matrix Step size, GAMS	$\mu_i(n+1) = \left[ \mu_i(n) + \rho_i \operatorname{Re} \left[ \hat{\Psi}_i^*(n) \mathbf{u}_i(n) e^*(n) \right] \right]_{\mu-}^{\mu+}$ $\hat{\Psi}_i(n+1) = [1 - \mu_i(n) \mathbf{q}_i(n) \mathbf{u}_i^*(n)] \hat{\Psi}_i(n) + \mathbf{g}_i(n)$
GASS with $\alpha$ , GASS- $\alpha$	$\mu(n+1) = \left[ \mu(n) + \rho \operatorname{Re} \left[ \hat{\Psi}^H(n) \mathbf{u}(n) e^*(n) \right] \right]_{\mu-}^{\mu+}$ $\hat{\Psi}(n+1) = \alpha \hat{\Psi}(n) + \mathbf{g}(n)$
GAMS with $\alpha$ , GAMS- $\alpha$	$\mu_i(n+1) = \left[ \mu_i(n) + \rho_i \operatorname{Re} \left[ \hat{\Psi}_i^*(n) \mathbf{u}_i(n) e^*(n) \right] \right]_{\mu-}^{\mu+}$ $\hat{\Psi}_i(n+1) = \alpha \hat{\Psi}_i(n) + \mathbf{g}_i(n)$
GASS- $\alpha$ with signum GASS- $\alpha$ -S	$\mu(n+1) = \left[ \mu(n) + \rho \operatorname{Re} \left[ \hat{\Psi}^H(n) \operatorname{sign}(\mathbf{u}(n)) e^*(n) \right] \right]_{\mu-}^{\mu+}$ $\hat{\Psi}(n+1) = \alpha \hat{\Psi}(n) + \mathbf{g}(n)$
GAMS- $\alpha$ with signum GAMS- $\alpha$ -S	$\mu_i(n+1) = \left[ \mu_i(n) + \rho_i \operatorname{Re} \left[ \hat{\Psi}_i^*(n) \operatorname{sign}(\mathbf{u}_i(n)) e^*(n) \right] \right]_{\mu-}^{\mu+}$ $\hat{\Psi}_i(n+1) = \alpha \hat{\Psi}_i(n) + \mathbf{g}_i(n)$

Among VSS methods, the step size update method in [13] performs relatively better than the others. Its superiority comes from the statistic used to update the step size. Below, the slightly modified version of this method is demonstrated as the VSS method:

$$\mu(n) = \left[ \theta \mu(n-1) + \gamma |p(n)|^m \right]_{\mu-}^{\mu+} \quad (3.30)$$

$$p(n) = \beta p(n-1) + (1-\beta)e(n)e(n-1) \quad (3.31)$$

where  $m$  equals 1 or 2,  $\gamma > 0$ ,  $\beta$  and  $\theta$  are less and close to 1. This VSS method gives better indication of divergence from the optimum solution by partially removing the effect of white measurement noise term in the estimation error  $e(n)$ . Especially, the parameters  $\theta$  and  $\gamma$  determine the steady state value of the step size and the sensitivity to instantaneous changes in the optimum solution.

The main disadvantage of the VSS methods is that; even the optimum step size is obtained by adjusting design parameters for an adaptive filtering problem, whenever a change occurs such that optimum step size value changes, VSS method may respond the change in the wrong way. That is, while an increase is required to obtain the new optimum step size value, the VSS may introduce a decrease in the step size, hence the steady state performance becomes worse than the before. The contradictory behavior of the VSS may cause misadjustment even larger than that of the zero forcing step size. This may happen when a step size value that is much less than the optimum step size is attained by the VSS. This behavior stems from different slopes of the MSE performance index at the two sides of the optimum step size. Some negative deviation from the optimum step size causes more increase in the cost function (MSE) than the positive deviation from the optimum step size. Despite its drawback, the VSS method has one advantage over the gradient adaptive step size method. The VSS method in (3.30) and (3.31) has considerably less computational complexity than that of the gradient adaptive step size method and its simplified versions. Therefore, the VSS methods are preferred over GAS methods, when optimum solution is approximately time invariant and there is not an optimum step size value required for tracking. In the thesis, one such application, namely the acoustic echo cancellation, is simulated. In acoustic echo cancellation, changes in the optimum solution are considered to be the instantaneous changes.

#### 4. EXPONENTIALLY WEIGHTED AFFINE PROJECTION ALGORITHM

In this section, we propose the Exponentially Weighted version of the APA (EW-APA), which is very similar to the Recursive Least Squares (also known as Exponentially Weighted RLS, EW-RLS) algorithm. They have similar estimation methods of the inverse time averaged correlation matrix. They use the estimate of the inverse exponentially weighted correlation matrix. The RLS algorithm uses an  $M \times M$  one [1], whereas EW-APA(P) uses a  $P \times P$  one. Using this matrix provides smaller misadjustment, but the convergence speed becomes slower, especially when the optimum solution changes instantaneously.

The EW-APA uses the estimate of the inverse of the exponentially weighted correlation matrix, which is calculated recursively as follows:

$$\hat{\Phi}^{-1}(n) = \lambda^{-1} \hat{\Phi}^{-1}(n-1) + \frac{\lambda^{-1} \hat{\Phi}^{-1}(n-1) \mathbf{u}_p(n) \mathbf{u}_p^H(n) \hat{\Phi}^{-1}(n-1)}{\lambda + \mathbf{u}_p^H(n) \hat{\Phi}^{-1}(n-1) \mathbf{u}_p(n)} \quad (4.1)$$

where  $\lambda$  is the forgetting factor and  $\mathbf{u}_p(n)$  is the input vector  $\mathbf{u}(n)$  with the first  $P$  elements. The regularization is only needed at the initialization,  $\hat{\Phi}^{-1}(0) = \delta^{-1} \mathbf{I}$ . Equation (4.1) is obtained from the estimate of the exponentially weighted correlation matrix in (4.2) by using the matrix inversion lemma.

$$\hat{\Phi}(n) = \lambda \hat{\Phi}(n-1) + \mathbf{u}_p(n) \mathbf{u}_p^H(n) \quad (4.2)$$

Employing the estimate of the inverse correlation matrix or the zero forcing projection matrix in the algorithms bring some advantages and disadvantages depending on the application. The RLS algorithm and the EW-APA may provide both fast convergence and small misadjustment, when the data pre-windowing assumption is valid. That is to say: the input process is equal to zero (leading zeros) before the algorithm starts to operate. But,

when the optimum solution changes instantaneously, this is not valid and these algorithms converge too slowly.

The zero forcing in the LS algorithm and the APA may provide convergence in filter length ( $M$ ) iterations; however they do not provide good steady state performance due to the noise amplification. Employing the inverse correlation matrix instead of the zero forcing matrix provides better steady state performance, but then the convergence speed gets slower due to loss of the zero forcing. The RLS algorithm is a typical example of this problem.

The EW-APA has step size of one; hence it provides approximately the zero forcing convergence speed. The forgetting factor is more important than the step size, because some range of values may lead to divergence of the adaptive filter. There may be an optimum value of forgetting factor, which provides minimum misadjustment with the nonstationary optimum solution. Regarding this, the gradient adaptive forgetting factor (GAFF) RLS algorithms are also proposed [16]. The EW-RLS algorithm has a divergence problem when some range of the forgetting factor value is used. Divergence problem must be regarded by using strict lower bound for forgetting factor in the GAFF update. Usually, the bounds and the optimum value are too close and large fluctuations in the stochastic gradient update of the forgetting factor prevent significant steady state performance improvement with the GAFF. Therefore, the GAFF provides superiority only with very short filter lengths.

The deviations from the optimum forgetting factor value cause larger misadjustment. The greater forgetting factor value makes the memory of the algorithm (defined as  $1/(1 - \lambda)$ ) in the RLS literature [1]) longer than the required for the optimum tracking. Inversely, the smaller value makes the memory of the algorithm shorter. Although, both types of deviation lead to greater misadjustment, first one (long memory) causes more increase in misadjustment than the second one.

Using near optimum forgetting factor value and step size of one, the EW-APA can provide minimum misadjustment similar to the GAS-APA. But, it exhibits slow convergence when an instantaneous change occurs in the optimum solution. This trade off

problem may be partially removed by using the GAS with the EW-APA. This can be achieved by using a forgetting factor value smaller than the optimum value. This value may provide a convergence speed that is near to that of the APA. However, this forgetting factor value may not provide minimum misadjustment with the EW-APA. Using the GAS-EW-APA, the minimum misadjustment may be obtained. When the optimal forgetting factor is used, the mean step size of the GAS settles to one, (making GAS useless). Forgetting factor values that are greater than the optimum value also cause mean step size value of approximately one. The misadjustment becomes larger than the minimum and the convergence speed gets slower than that of the optimum forgetting factor.

#### 4.1. Switching Exponentially Weighted Affine Projection Algorithm

We know that using the estimate of the exponentially weighted correlation matrix provides small misadjustment, but also causes convergence speed loss. To obtain both convergence speed of the APA and small misadjustment of the EW-APA, we propose a switching method such that the zero forcing matrix is used in convergence and the estimate of the inverse correlation matrix is used in steady state. We call this algorithm the Switching Exponentially Weighted Affine Projection Algorithm (SEW-APA). The SEW-APA has the fast convergence property of the APA and much smaller misadjustment than that of the APA. To obtain the minimum misadjustment, the use of the optimum forgetting factor is needed. The switching process requires the detection of the convergence. It is achieved by using the same method of the VSS in (3.30) and (3.31). The SEW-APA equations can be written as:

$$\mathbf{e}_p(n) = \mathbf{d}_p(n) - \mathbf{U}_p^T(n)\mathbf{w}^*(n) \quad (4.3)$$

$$\hat{\Phi}_c^{-1}(n) = \lambda^{-1}\hat{\Phi}_c^{-1}(n-1) + \frac{\lambda^{-1}\hat{\Phi}_c^{-1}(n-1)\mathbf{u}_p(n)\mathbf{u}_p^H(n)\hat{\Phi}_c^{-1}(n-1)}{\lambda + \mathbf{u}_p^H(n)\hat{\Phi}_c^{-1}(n-1)\mathbf{u}_p(n)} \quad (4.4)$$

$$\Phi_z(n) = \mathbf{U}_p^H(n)\mathbf{U}_p(n) \quad (4.5)$$

$$p(n) = \beta p(n-1) + (1-\beta)e(n)e(n-1) \quad (4.6)$$

When  $|p(n)|^m > \text{threshold}$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mathbf{U}_p(n)\hat{\Phi}_z^{-1}(n)\mathbf{e}_p^*(n) \quad (4.7)$$

otherwise

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mathbf{U}_p(n) \hat{\Phi}_c^{-1}(n) \mathbf{e}_p^*(n) \quad (4.8)$$

where  $m$  equals one or two,  $\beta$  is less and close to one, the subscripts  $z$  and  $c$  have the meanings of the zero forcing and the inverse correlation, respectively.  $e(n)$  is the a priori estimation error.

To obtain the minimum or near minimum misadjustment, the SEW-APA requires the optimum or near optimum forgetting factor values when the optimum solution is nonstationary. This is the major drawback of the algorithm compared to the GAS-APA.

## 5. COMPUTER SIMULATIONS

In this section, we demonstrate the performances of the proposed algorithms with computer simulation results. The nonstationary system identification problem that requires optimum parameter values to obtain the minimum misadjustment is simulated to demonstrate the tracking performance of the GAS-APA. The acoustic echo cancellation is simulated as an adaptive filtering application with long filter lengths, in which the optimum solution is assumed to be stationary and there might be instantaneous changes in the optimum solution. Finally, the nonstationary inverse system identification problem is simulated to demonstrate the tracking performances for that problem.

### 5.1. Nonstationary System Identification

In the system identification problem, the input signal of the unknown system is also applied to the input of the adaptive filter. The adaptive filtering algorithm adjusts the parameters of the filter in order to model the unknown parameters of the system. The system identification configuration with an adaptive filter is shown in Figure 5.1.

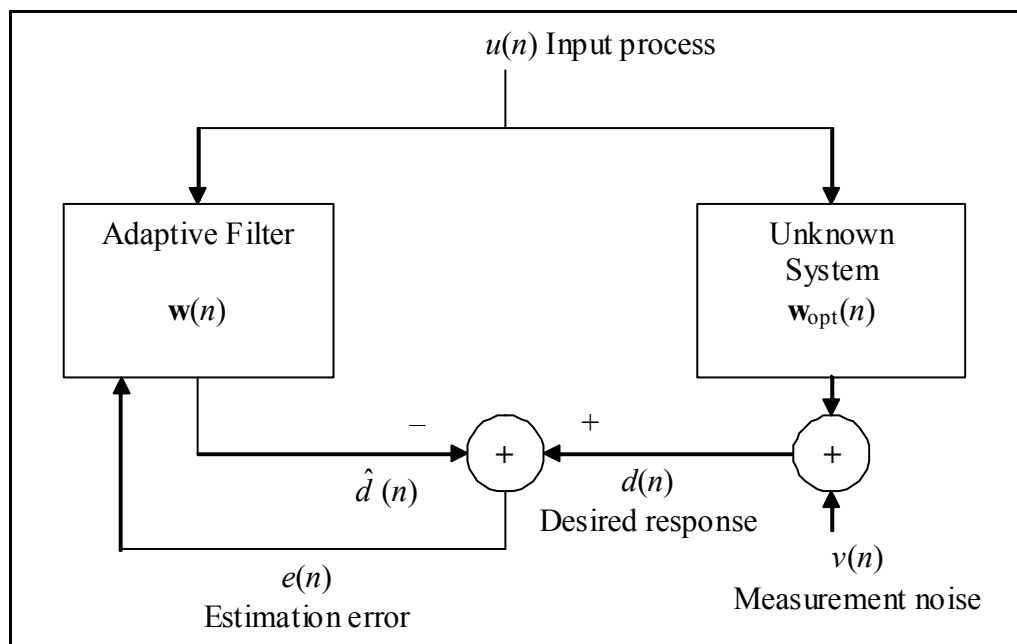


Figure 5.1. System identification configuration

The unknown system parameters (the optimum solution;  $\mathbf{w}_{opt}$ ) may be time invariant (stationary) or time varying (nonstationary). For the nonstationary system, the adaptive filtering algorithm must track the time-varying optimum solution. The GAS-APA is designed to provide both tracking of the optimum solution and fast convergence.

A model for the unknown time varying system is the first-order Markov model [1]. In this model, the unknown system is modeled as a transversal filter whose tap-weight vector  $\mathbf{w}_{opt}(n)$  undergoes a first-order Markov process, which could be written in vector form as:

$$\mathbf{w}_{opt}(n+1) = a \mathbf{w}_{opt}(n) + \boldsymbol{\omega}(n) \quad (5.1)$$

where  $a \leq 1$ , and  $\boldsymbol{\omega}(n)$  is the process noise vector having zero mean and the correlation matrix  $\mathbf{Q}$ . The process noise vector  $\boldsymbol{\omega}(n)$  is independent of both the input vector  $\mathbf{u}(n)$  and the measurement noise  $v(n)$ .

When no a priori knowledge of changes in the unknown system is available, the random walk model is employed to model the nonstationary system. The random walk model is obtained by setting  $a$  equal to one in the first order Markov model [1].

In the thesis, the random walk model is used to model the nonstationary unknown system. The process noise vector  $\boldsymbol{\omega}(n)$  will be a zero mean, white Gaussian process vector with a diagonal autocorrelation matrix  $\mathbf{Q} = \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_M^2)$ . The variance values  $\sigma_i^2$  are set to different values to demonstrate the superiority of the gradient adaptive matrix step size (GAMS) to the gradient adaptive scalar step size (GASS).

In the nonstationary system identification problem, the input process  $u(n)$  is chosen as a highly correlated, low-pass, AR(1) process obtained by filtering a white Gaussian process of zero mean and unit variance with the correlating filter:

$$H_{cor}(z) = \frac{0.5}{1 - 0.95z^{-1}} \quad (5.2)$$

The optimum solution  $\mathbf{w}_{opt}(n)$  has length of 10, and the adaptive filter has also the same length. The random walk model having different variance values for the process noise vector is used to implement time variations in the optimum solution  $\mathbf{w}_{opt}(n)$ . The variance values  $\sigma_i^2$  are equal to  $10^{-4}$  for the first three tap-weights ( $i = 1$  to 3) and equal to  $10^{-6}$  for the others. The initial vector  $\mathbf{w}_{opt}(0)$  is obtained from a zero mean, white Gaussian process vector having its first three entries of variance one and the others of variance  $9.0 \times 10^{-2}$ . A white Gaussian process having zero mean and variance  $\sigma_v^2$  of  $10^{-2}$  is added to the output of the unknown system as the measurement noise. At iteration  $20.0 \times 10^3$ , an instantaneous change to the optimum solution is applied by setting it equal to the negative of the start up value ( $\mathbf{w}_{opt}(20.000) = -\mathbf{w}_{opt}(0)$ ). After that instance, the measurement noise variance  $\sigma_v^2$  is increased to  $4.0 \times 10^{-2}$ , hence the optimum step size value and the minimum misadjustment change. The eigenvalue spread of the  $10 \times 10$  autocorrelation matrix  $\mathbf{R}$  of the input process  $u(n)$  is approximately 328 ( $\lambda_{\max} = 22.0$   $\lambda_{\min} = 0.067$ ).

Before proceeding to the simulation results, the parameter values for the algorithms are stated. All parameter values are determined with trial and error in the thesis.

- NLMS algorithm and APA(2) have  $\mu = 1$ .
- VSS-NLMS algorithm has;  $m = 2$ ,  $\gamma = 50$ ,  $\beta = 0.99$  and  $\theta = 0.99$ .
- VSS-APA(2) has;  $m = 2$ ,  $\gamma = 50$ ,  $\beta = 0.99$  and  $\theta = 0.97$ .
- GASS-NLMS algorithm has  $\rho = 0.04$ .
- GASS-APA(2) has  $\rho = 0.05$ .
- GASS- $\alpha$ -NLMS algorithm has  $\alpha = 0.99$  and  $\rho = 0.02$ .
- GASS- $\alpha$ -APA(2) has  $\alpha = 0.99$  and  $\rho = 0.02$ .
- GASS- $\alpha$ -S-NLMS algorithm has  $\alpha = 0.99$  and  $\rho = 0.02$ .
- GASS- $\alpha$ -S-APA(2) has  $\alpha = 0.99$  and  $\rho = 0.02$ .
- GAMS-NLMS algorithm has  $\rho = 0.5$ .
- GAMS-APA(2) has  $\rho = 0.05$ .
- GAMS- $\alpha$ -NLMS algorithm has  $\alpha = 0.99$  and  $\rho = 0.1$ .
- GAMS- $\alpha$ -APA(2) has  $\alpha = 0.99$  and  $\rho = 0.05$ .
- GAMS- $\alpha$ -S-NLMS algorithm has  $\alpha = 0.99$  and  $\rho = 0.2$ .
- GAMS- $\alpha$ -S-APA(2) has  $\alpha = 0.99$  and  $\rho = 0.1$ .

- EW-APA(2) has  $\mu = 1$ ,  $\lambda$  values of: 0.995, 0.95 and 0.98 ( $\lambda_{opt} \approx 0.98$ ).
- GASS- $\alpha$ -S-EW-APA(2) has  $\lambda = 0.95$  and  $\rho = 0.04$ .
- SEW-APA(2) has  $\lambda = 0.99$ ,  $m = 2$ ,  $\beta = 0.99$ ,  $threshold = 10^{-5}$ .
- The algorithms use a regularization parameter value of  $10^{-2}$  ( $\delta = 10^{-2}$ ).
- The step size is bounded with  $\mu_{max} = 1$  and  $\mu_{min} = 10^{-2}$  for the GAS and the VSS algorithms.

In the simulation results, the misadjustment values demonstrate the steady state performances. The mean step size values in the steady state will be given for the VSS and the GAS algorithms. The convergence speeds are determined by using the Mean Square Deviation (MSD) curves of the algorithms. MSD is defined as:

$$D(n) = E[\|\mathbf{w}(n) - \mathbf{w}_{opt}(n)\|^2] \quad (5.3)$$

The convergence speed is expressed as the number of iterations required to converge “close enough” to the optimum solution. The results are obtained by ensemble-averaging 10 independent runs. In Figure 5.2, the MSD curves for the APA(2) and the GASS- $\alpha$ -S-APA(2) are given as an example of the convergence characteristics. In Figure 5.3, the step size curve of the GASS- $\alpha$ -S-APA(2) is plotted to demonstrate a step size update example. Steady state performances are evaluated by using the misadjustment ratio ( $M$ ) which is defined by:

$$M = \frac{J(\infty) - J_{min}}{J_{min}} \quad (5.4)$$

$$J(n) = E[|e(n)|^2] \quad (5.5)$$

$$J_{min} = \sigma_v^2 \quad (5.6)$$

Firstly, the simulation results for the VSS and the GAS NLMS algorithms are tabulated in Table 5.1. In the table heading,  $M$ ,  $CS$  and  $E[\mu(n)]$  stand for the misadjustment, the convergence speed and the mean step size, respectively.

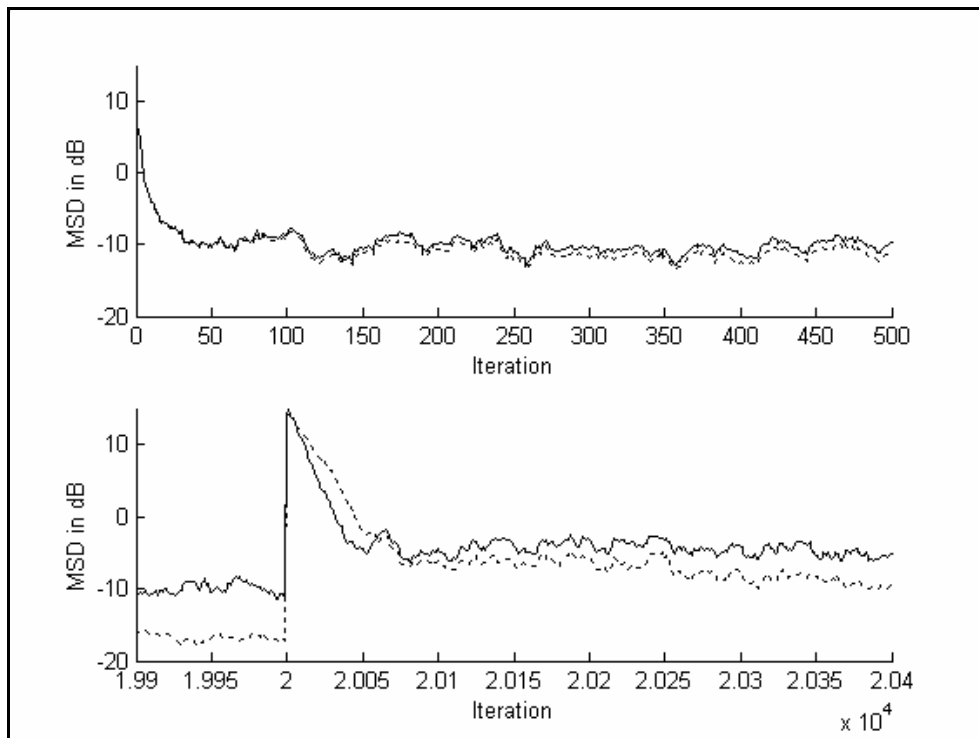


Figure 5.2. MSD curves for the APA(2) and the GASS- $\alpha$ -S-APA(2) as an example of the convergence characteristics

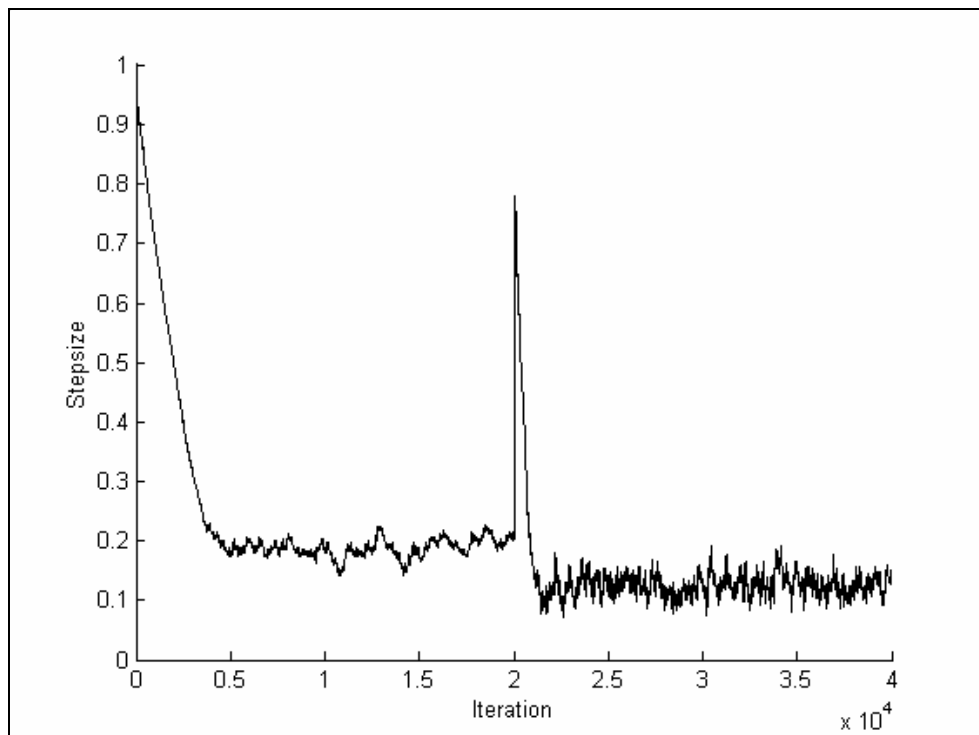


Figure 5.3. The step size curve for the GASS- $\alpha$ -S-APA(2)

Table 5.1. Simulation results for the VSS and the GAS NLMS algorithms

Algorithm	$n$ from 1 to 19999			$n$ from 20000 to 40000		
	$M$ (%)	$CS$ (ite.)	$E[\mu(n)]$	$M$ (%)	$CS$ (ite.)	$E[\mu(n)]$
NLMS (APA(1))	170	250	1	138	250	1
VSS-NLMS	150	250	0.15	113	250	0.75
GASS-NLMS	90	250	0.4	43	250	0.2
GASS- $\alpha$ -NLMS	100	250	0.3	45	250	0.18
GASS- $\alpha$ -S-NLMS	100	250	0.3	45	250	0.18
GAMS-NLMS	73	250	0.55 0.15	43	600	0.35 0.2
GAMS- $\alpha$ -NLMS	70	250	0.55 0.1	35	600	0.35 0.12
GAMS- $\alpha$ -S-NLMS	70	250	0.55 0.1	35	600	0.35 0.12

The simulation results demonstrate that the convergence speeds of the GASSs and the VSS NLMS algorithms are approximately the same as that of the NLMS algorithm, which has step size value of one. However, the GAMSs NLMS algorithms do not have the convergence speed of the NLMS algorithm, when an instantaneous change occurs in the optimum solution. Because, the step size update of the GAMS methods do not well respond to the change and the step size values do not reach to one. The reason of this behavior is that: the scalar gradient ( $\hat{\nabla}_{\mu_i}(n)$ ) used in the step size update equation of the GAMS is a stochastic gradient. In the scalar step size update (GASS), the scalar gradient  $\hat{\nabla}_{\mu}(n)$  is also stochastic, but it is averaged due to the existence of the vector dot product in (3.22). Therefore, the GASS does not show the inefficient step size update of the GAMS. However, later it will be shown that the GAMS-APA( $P>1$ ) does not have the same

inefficient step size update of the GAMS-NLMS algorithm, because the scalar gradient ( $\hat{\nabla}_{\mu_i}(n)$ ) is obtained by using the elements of the projected gradient vector ( $\mathbf{g}_i(n)$ ) when  $P$  is greater than one.

When the steady state performances are examined in Table 5.1, it is observed that the GAS methods provide much smaller misadjustment than that of the NLMS algorithm. The VSS-NLMS algorithm can not provide that much improvement, because it can not obtain and track the optimum step size. The GASS-NLMS algorithm obtains the optimum scalar step size value; hence it provides the minimum misadjustment. The simplified GASS methods have smaller mean step size values than the optimum; hence they provide slightly larger misadjustment than that of the GASS. The GAMS NLMS algorithms obtain the optimum mean matrix step size values, which provide smaller misadjustment than that of the GASS. Since some of the tap-weight parameters in the optimum solution (first three) change faster, they require different optimum step size values. The GAMS methods can provide these optimum step size values. In Table 5.1, the mean step size values of the GAMS methods are given. The first value is the mean step size for the first three tap-weights, and the second one is for the others. The simplified GAMS methods provide more accurate optimum step sizes than the GAMS because of the averaging provided by the scalar multiplier  $\alpha$  in the step size update. The GAMS just calculates the instantaneous estimates, but the scalar  $\alpha$  provides an averaging (averaging exists in the theoretical derivation of the GAMS). Therefore, the simplified versions of the GAMS are superior to the GAMS for the NLMS algorithm. However, as we will see below, the GAMS method is superior to the simplified GAMS methods, when the projection order is larger than one.

Now, the simulation results for the VSS and the GAS APA(2) are tabulated in Table 5.2. The simulation results demonstrate that the VSS, the GASSs and the GAMSs APA(2) can provide the convergence speed of the APA(2), even when the optimum solution changes instantaneously. The VSS-APA(2) algorithm can not provide the minimum misadjustment, because it can not obtain the optimum step size. However, the GAS methods provide the optimum step size values and the minimum misadjustment. The GASS methods have approximately the same mean step size values. This shows that the simplified GASS methods can also provide the optimum mean scalar step size for higher

projection orders than one. However, more significant deviations had occurred for the simplified GASS NLMS algorithms. The difference could be explained with the fact that the GASS APA( $P > 1$ ) uses a projected gradient vector in the step size and the tap-weight updates, whereas GASS NLMS algorithms use a totally stochastic gradient vector.

Table 5.2. Simulation results for the VSS and the GAS APA(2)

Algorithm	$n$ from 1 to 19999			$n$ from 20000 to 40000		
	$M$ (%)	$CS$ (ite.)	$E[\mu(n)]$	$M$ (%)	$CS$ (ite.)	$E[\mu(n)]$
APA(2)	300	50	1	295	50	1
VSS-APA(2)	120	50	0.1	145	50	0.38
GASS-APA(2)	80	50	0.2	45	50	0.15
GASS- $\alpha$ -APA(2)	80	50	0.2	45	50	0.15
GASS- $\alpha$ -S-APA(2)	80	50	0.2	43	50	0.13
GAMS-APA(2)	80	50	0.30 0.15	60	50	0.32 0.21
GAMS- $\alpha$ -APA(2)	80	50	0.33 0.15	78	50	0.35 0.28
GAMS- $\alpha$ -S-APA(2)	80	50	0.32 0.13	73	50	0.33 0.25

The GASS methods for the APA(2) are superior to the GAMS counterparts. The high performance of the GASS in tracking could be explained with the existence of the dot vector product in the step size update of the GASS and the averaging provided by this operation. Therefore, the GASS methods obtain the scalar optimum step size better than the GAMS methods do.

We could also state that the GAMS methods do not give much misadjustment improvement over the GASS methods despite their higher computational complexity. Moreover, the GAMS methods provide slower convergence to the optimum step size than the GASS counterparts. The GAMS-NLMS algorithms also fail to give acceptable convergence with the instantaneous change of the optimum solution. Regarding these issues, we could conclude that the GASS-APA is more preferable than the GAMS-APA.

In Table 5.3, we demonstrate the performances of the EW-APA(2), the GASS- $\alpha$ -S-EW-APA(2) and the SEW-APA(2).

Table 5.3. Simulation results for the EW-APA(2), the GASS- $\alpha$ -S-EW-APA(2) and the SEW-APA(2)

Algorithm	$n$ from 1 to 19999			$n$ from 20000 to 40000		
	$M$ (%)	$CS$ ite.	$E[\mu(n)]$	$M$ (%)	$CS$ ite.	$E[\mu(n)]$
EW-APA(2) $\lambda=0.95$	120	50	1	93	50	1
EW-APA(2) $\lambda=0.98$	75	50	1	38	200	1
EW-APA(2) $\lambda=0.99$	95	50	1	33	370	1
GASS- $\alpha$ -S-EW-APA $\lambda = 0.95$	75	50	0.42	35	120	0.28
SEW-APA(2) $\lambda=0.99$	95	50	1	33	50	1

The simulation results demonstrate that the convergence speed of the EW-APA gets slower with increasing forgetting factor, especially when instantaneous change occurs. Some relatively small forgetting factor values (here 0.95) might provide nearly convergence speed of the APA. The SEW-APA(2) detects the convergence and uses the APA(2), hence it has the convergence speed of the APA(2). The amount of the misadjustment is dependent on the value of the forgetting factor. In this simulation, the optimum forgetting factor is 0.98 for the first stage and it provides 75 per cent

misadjustment. The GAS-EW-APA can also provide this misadjustment by properly assigning the forgetting factor (here 0.95). On the other hand, the convergence speed gets slower than that of the EW-APA having forgetting factor of 0.95. There is still a trade off problem. We also observe that the EW methods are more robust to the measurement noise and they provide smaller misadjustment (as small as 33 per cent) than that of the GAS methods (the smallest is obtained with GASS- $\alpha$ -S-APA(2); 43 per cent). However, the convergence speed loss is the problem of the EW-APA. The SEW-APA(2) may solve this problem and provide both fast convergence and small misadjustment.

## 5.2. Steady State Analysis of Affine Projection Algorithm

In this section, the steady state performance analysis of the APA is derived. For the nonstationary system identification model defined above, the MSE and the misadjustment are calculated. The numerical and experimental results are compared to prove the validity of the analysis.

The MSE cost function in (A.26) and (A.29) can also be proven to be valid for the APA. Below, we will derive the excess MSE of the APA for the random walk model nonstationary system identification.

Firstly, we rewrite the system equations required to derive the MSD and the excess MSE.

$$\mathbf{d}_p(n) = \mathbf{U}_p^T(n) \mathbf{w}_{opt}^*(n) + \mathbf{v}(n) \quad (5.7)$$

$$\mathbf{e}_p(n) = \mathbf{d}_p(n) - \mathbf{U}_p^T(n) \mathbf{w}^*(n) \quad (5.8)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \mathbf{U}_p(n) \hat{\mathbf{\Phi}}^{-1}(n) \mathbf{e}_p^*(n) \quad (5.9)$$

$$\mathbf{w}_{opt}(n+1) = \mathbf{w}_{opt}(n) + \boldsymbol{\omega}(n) \quad (5.10)$$

$$\boldsymbol{\epsilon}(n) = \mathbf{w}(n) - \mathbf{w}_{opt}(n) \quad (5.11)$$

Using these equations, we could write a recursion for the weight error vector as:

$$\boldsymbol{\epsilon}(n+1) = [\mathbf{I} - \mu \mathbf{U}_p \hat{\mathbf{\Phi}}^{-1} \mathbf{U}_p^H] \boldsymbol{\epsilon}(n) + \mu \mathbf{U}_p \hat{\mathbf{\Phi}}^{-1} \mathbf{v}^*(n) - \boldsymbol{\omega}(n) \quad (5.12)$$

Multiplying both sides of (5.12) with its hermitian, we get

$$\begin{aligned} \boldsymbol{\epsilon}(n+1)\boldsymbol{\epsilon}^H(n+1) &= [\mathbf{I} - \mu\mathbf{U}_p\hat{\boldsymbol{\Phi}}^{-1}\mathbf{U}_p^H]\boldsymbol{\epsilon}(n)\boldsymbol{\epsilon}^H(n)[\mathbf{I} - \mu\mathbf{U}_p\hat{\boldsymbol{\Phi}}^{-1}\mathbf{U}_p^H] \\ &\quad + \mu^2\mathbf{U}_p\hat{\boldsymbol{\Phi}}^{-1}\mathbf{v}\mathbf{v}^T\hat{\boldsymbol{\Phi}}^{-1}\mathbf{U}_p^H + \boldsymbol{\omega}\boldsymbol{\omega}^H \end{aligned} \quad (5.13)$$

To obtain the excess MSE from (5.13), we multiply both sides with  $\mathbf{U}_p\mathbf{U}_p^H$  and then apply the expectation ( $E[\cdot]$ ) and the trace ( $\text{tr}[\cdot]$ ) operators to both sides. Finally we set iteration index  $n$  to infinity ( $n \rightarrow \infty$ ). Now, some terms cancel each other and some vanish to zero by employing independence assumptions [1]. We get

$$2\mu\text{tr}[\mathbf{R}\mathbf{K}(\infty)] - \mu^2\text{tr}[\mathbf{R}\mathbf{K}(\infty)E[\mathbf{U}_p\hat{\boldsymbol{\Phi}}^{-1}\mathbf{U}_p^H]] = P\mu^2\sigma_v^2 + \text{tr}[\mathbf{R}\mathbf{Q}] \quad (5.14)$$

where  $\mathbf{K}(n)$  is the correlation matrix of the weight error vector  $\boldsymbol{\epsilon}(n)$  (stated in (A.20)),  $\mathbf{Q}$  is the correlation matrix of the process noise vector  $\boldsymbol{\omega}(n)$  and  $\mathbf{R}$  is the autocorrelation matrix of the input process  $u(n)$ .

To be able to tract (5.14), the second term in the left side of the equation is approximated as

$$\text{tr}[\mathbf{R}\mathbf{K}(\infty)E[\mathbf{U}_p\hat{\boldsymbol{\Phi}}^{-1}\mathbf{U}_p^H]] = \kappa \text{tr}[\mathbf{R}\mathbf{K}(\infty)] \quad (5.15)$$

where  $\kappa$  is a positive scalar and it depends on the order  $P$  and the step size  $\mu$  ( $\kappa = \mathcal{F}(P, \mu)$ ).

Now, we can write the excess MSE by using (5.14) and (5.15):

$$\begin{aligned} J_{exc}(\infty) &= \text{tr}[\mathbf{R}\mathbf{K}(\infty)] \\ &= \frac{P\sigma_v^2\mu}{2 - \mu\kappa} + \frac{\text{tr}[\mathbf{R}\mathbf{Q}]}{2\mu - \mu^2\kappa} \end{aligned} \quad (5.16)$$

where the first term is the estimation variance due to the measurement noise and the second term is the lag variance due to the process noise vector  $\boldsymbol{\omega}(n)$ .

The excess MSE defined in (5.16) takes its minimum value when the two terms in the addition are equal. The optimum step size value that provides the minimum MSE can be written as

$$\mu_{opt} = \sqrt{\frac{\text{tr}[\mathbf{RQ}]}{P\sigma_v^2}} \quad (5.17)$$

The optimum step size value defined above is consistent with the simulation result of the GASS-APA(2). When we calculated the optimum step size from (5.17) for the first stage of simulation, we get 0.2. This value is same as the mean step size in Table 5.2. Corresponding misadjustment is 80 per cent. The mean step size of the GASS-NLMS algorithm is equal to 0.4, although we calculate the optimum step size as 0.28. When we use 0.28 as step size for the NLMS algorithm, we obtain the minimum misadjustment (80 per cent) that is less than that of the GASS-NLMS (90 per cent).

We can also prove the validity of the approximation in (5.15). For the proof, we calculate  $\kappa$  by using results from a simulation in which nonstationarity is removed and the optimum step size is used. Then, the APA(2) gives 40 per cent misadjustment due to the first term in (5.16). The corresponding  $\kappa$  equals to 5. Using this value and the optimum step size in the minimum misadjustment equation:

$$M_{\min} = \frac{2 P \mu_{opt}}{2 - \mu_{opt} \kappa(P, \mu_{opt})} \quad (5.18)$$

we obtain the minimum misadjustment as 80 per cent, which is consistent with the simulation results for the GASS-APA(2).

### 5.3. Acoustic Echo Cancellation

The Acoustic Echo Cancellation (AEC) is an adaptive filtering system identification application. The acoustic echo cancellation problem differs from the system identification problem (defined above) in two main aspects: the input process  $u(n)$  is speech signal,

which is a highly correlated and nonstationary process, and the filter length is very long, ranging from a few hundreds to a few thousands tap-weights.

In Figure 5.4, a simple model of the acoustic echo cancellation is illustrated. This is a typical system identification configuration. The unknown system to be identified is the echo path, and it is also named the room impulse response. In the AEC simulations, the echo path is assumed to be stationary for a while, and any change is assumed to be an instantaneous one. Therefore, the VSS methods are preferred over the GAS methods due to their computational simplicity.

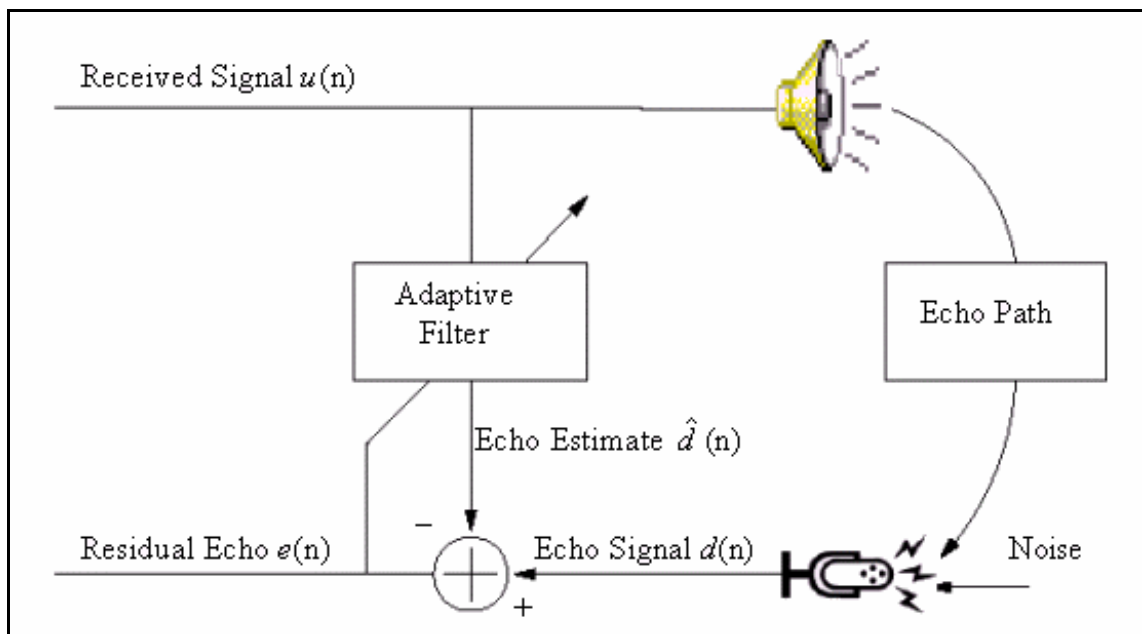


Figure 5.4. Acoustic echo cancellation configuration

The APA is widely used for the AEC, since it provides satisfactory convergence speed at reasonable computational complexity. However, the APA gives large misadjustment value when it is used with the step size value of one. Moreover, in the AEC problem, a double talk detector stops the tap-weight update of the adaptive filter, when the double talk starts. This process occurs with some delay. During this delay period, the double talk distorts the estimate of the tap-weight. This distortion becomes larger with the greater step size values [1,21]. Similarly, when the power of the input signal becomes weak, the estimate of the tap-weight is distorted again and the update must be stopped.

This is also detected with delay. When using a larger step size value, the distortion becomes larger. Using variable step size APA, smaller misadjustment (less residual echo), less double talk and no single talk distortions are obtained [1, 21, 22].

In the applications, the echo path delay may span from a few milliseconds (ms) to more than hundreds of milliseconds. When the input speech signal is sampled at 8 kHz, an echo path delay of 32 ms causes a room impulse response of 256 samples ( $8000 \times 0.032 = 256$ ). This small delay is suitable to model impulse response of small room, passenger car and telephone hybrid response (for Network Echo Cancellation) etc. A typical teleconference room could have delay spreads more than 250 ms and its room impulse response is more than 2000 samples at 8 kHz sampling rate. A room impulse response example is illustrated in Figure 5.5. This room impulse is used in the 256 ms AEC simulations of the thesis.

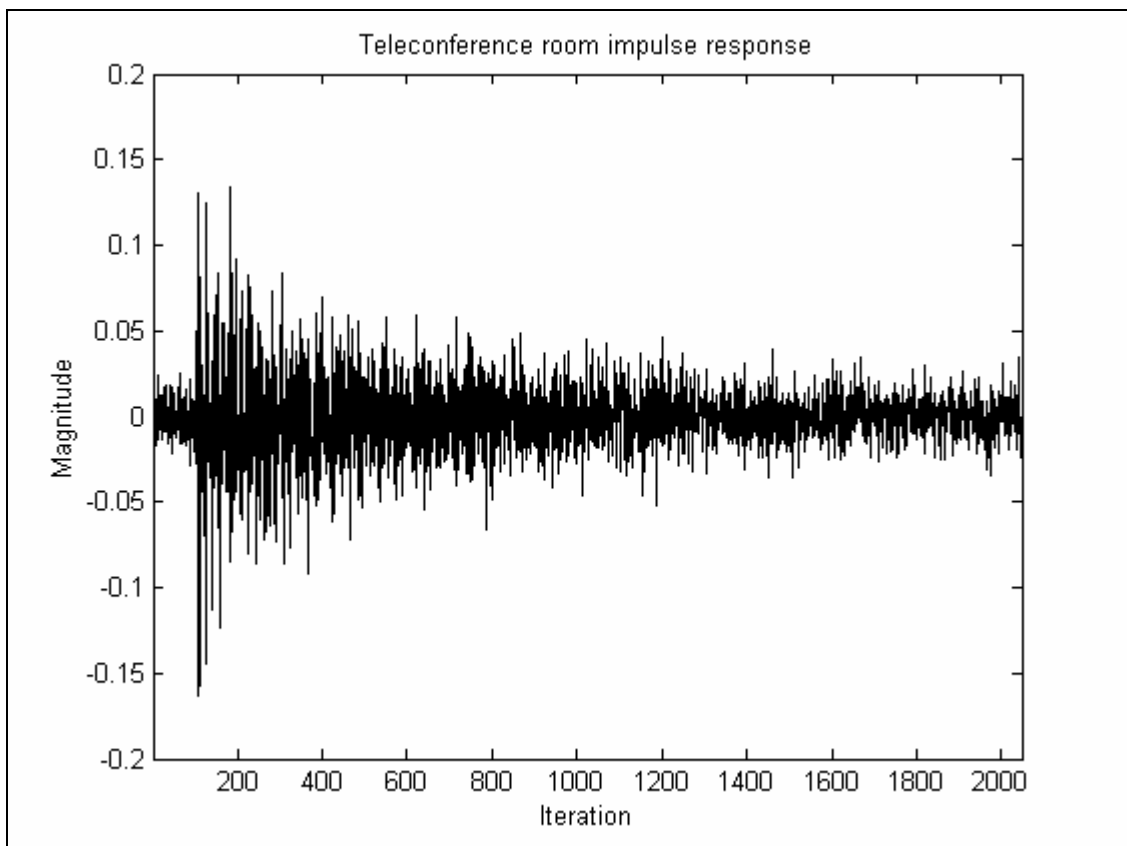


Figure 5.5. Teleconference room impulse response example

In the thesis, the AEC single talk simulations for 32 ms and 256 ms echo paths are performed. We will demonstrate the improvement provided by the VSS-APA. We will also give simulation results for the GASS- $\alpha$ -S and the SEW APA(2) for comparison.

In the single talk simulations, 8 kHz, 16-bit,  $-10$ dBm female speech is used as the input speech  $u(n)$ . A white Gaussian measurement noise of  $-50$  dBm is added to the echo signal  $d(n)$ . In the 32 ms and the 256 ms AEC simulation, a room impulse response and adaptive filter of length 256 and 2048, respectively, are used. In Figure 5.6, the input speech signal  $u(n)$  and its echo  $d(n)$  from the 256 ms echo path are plotted. An instantaneous change to the room impulse response is implemented by using its negative as the new room impulse response. At the same instant, measurement noise is also increased to  $-40$  dBm, hence behaviors of the algorithms with the instantaneous change are demonstrated.

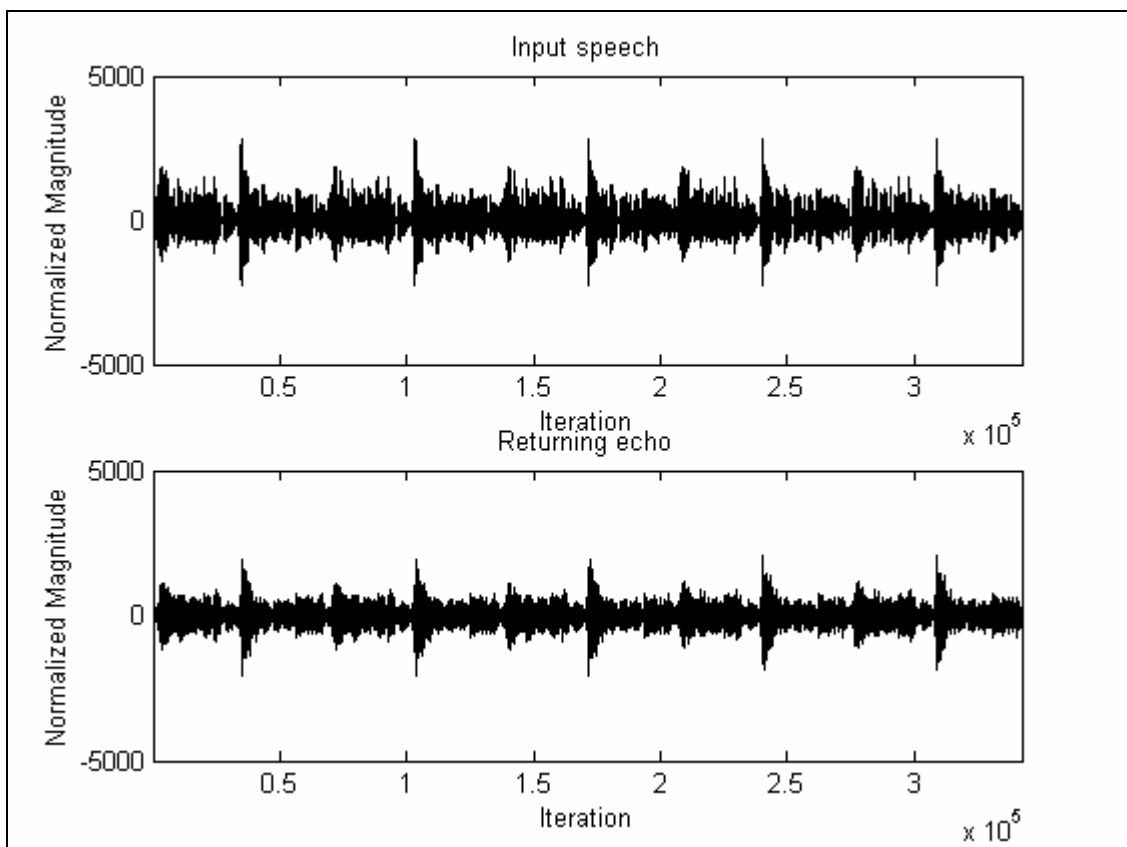


Figure 5.6. The input speech and its echo from the 256 ms echo path

Firstly, results of the 32 ms AEC simulation will be demonstrated. The NLMS algorithm and its VSS and GAS versions are used. The APA(2) could give slightly faster convergence than the NLMS algorithm, but it also has more computational cost. Because the NLMS gives satisfactory convergence speed for the filter length of 256, we will only simulate the NLMS algorithm.

In the 256 ms AEC simulation, a much longer echo path (256 ms teleconference room impulse response in Figure 5.5) and 2048 tap-weights adaptive filter are used. The NLMS algorithm can not provide satisfactory convergence speed with that long filter length, but the APA(2) can provide fast convergence.

For the AEC problems requiring long filter lengths, the subband adaptive filters can be used to get better performance than the full-band ones. The subband adaptive filtering provides the advantages of faster convergence, smaller misadjustment and lower computational complexity over the full-band adaptive filters. However, the delay introduced by the analysis and the synthesis filter banks emerges as the major drawback of the subband adaptive filtering. In the 256 ms AEC simulation, we will also use a subband adaptive filter that has 20 subbands, decimation rate of eight and the APA(2) in each subband. Each subband has an adaptive filter of length 256 (2048/8), operating at the decimated data rate of 1 ksps (8 ksps/8; decimation rate is eight).

The convergence and the steady state performances of the algorithms will be determined by using Echo Return Loss Enhancement (ERLE) curves. The convergence speed is determined as the time required (in seconds) to reach the mean ERLE. The mean value of the ERLE in the steady state will be used to evaluate the steady state performances. The ERLE is defined as:

$$\text{ERLE (dB)} = 10 \log_{10} \left( \frac{E[d^2(n)]}{E[e^2(n)]} \right) \quad (5.19)$$

In Figure 5.7, we demonstrate the square deviation curves for the NLMS and the VSS-NLMS algorithms in the 32 ms AEC simulation as an example of the convergence. In

Figure 5.8, we also demonstrate the ERLE curves of the same algorithms as an example of steady state performances.

Before proceeding to the simulation results, the parameter values for the algorithms are given:

- NLMS algorithm and fullband and subband APA(2) have  $\mu = 1$ .
- VSS-NLMS algorithm has  $m = 1$ ,  $\gamma = 10^{-4}$ ,  $\beta = 0.99$  and  $\theta = 0.99$ .
- GASS- $\alpha$ -S-NLMS algorithm has  $\alpha = 0.99$  and  $\rho = 2.0 \times 10^{-3}$ .
- VSS-APA(2) algorithm has;  $m = 2$ ,  $\gamma = 10^{-5}$ ,  $\beta = 0.99$  and  $\theta = 0.97$ .
- EW-APA(2) has parameters:  $\mu = 1$ ,  $\lambda = 0.9999$ .
- SEW-APA(2) has  $\lambda = 0.9999$ ,  $m = 1$ ,  $\beta = 0.9$ ,  $threshold = 10^2$ .
- The algorithms use a regularization parameter value of one ( $\delta = 1$ ).
- For the GAS and the VSS algorithms, the step size is bounded with  $\mu_{\max} = 1$  and  $\mu_{\min} = 10^{-1}$ .

In Table 5.4, we have the simulation results for the NLMS, the VSS-NLMS, and the GASS- $\alpha$ -S-NLMS algorithms in the 32 ms AEC simulation.

Table 5.4. Results of the 32 ms AEC simulation for the NLMS, the VSS-NLMS, and the GASS- $\alpha$ -S-NLMS algorithms

Algorithm	$n$ from 1 to inst. change			After inst. change		
	<i>ERLE</i> (dB)	<i>CS</i> (sec)	$E[\mu(n)]$	<i>ERLE</i> (dB)	<i>CS</i> (sec)	$E[\mu(n)]$
NLMS	24	0.3	1	13	0.3	1
VSS-NLMS	28	0.3	0.1	18	0.3	0.2
GASS- $\alpha$ -S-NLMS	27	0.3	0.2	17	0.3	0.2

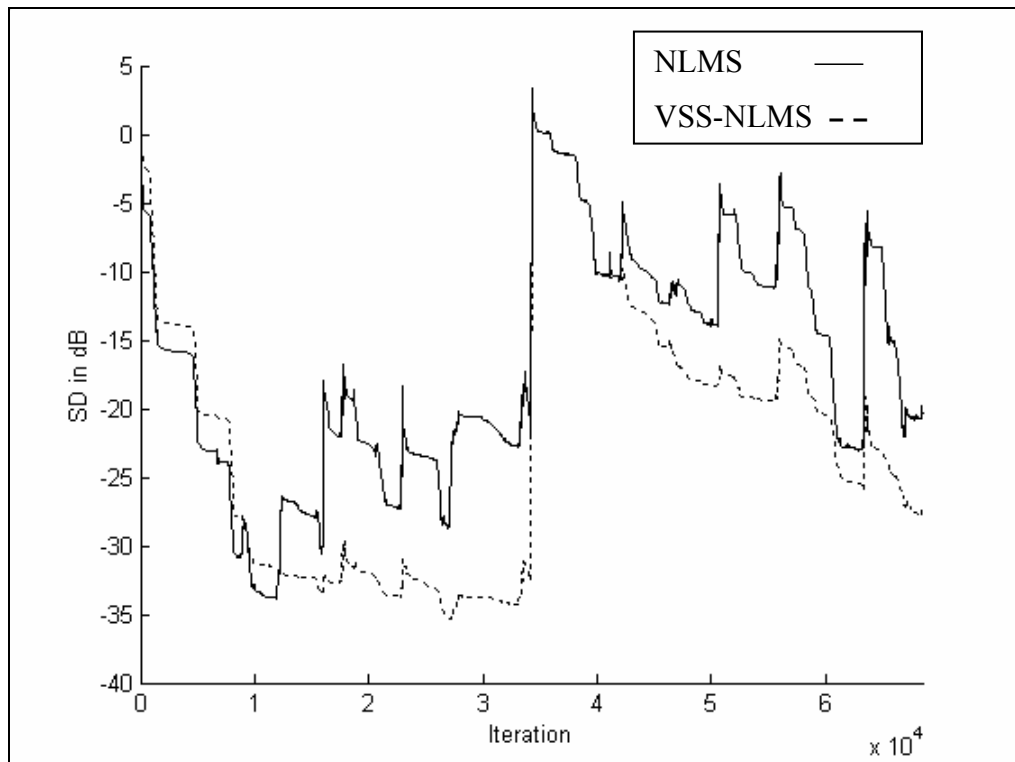


Figure 5.7. Square deviation curves for the NLMS and VSS-NLMS algorithms

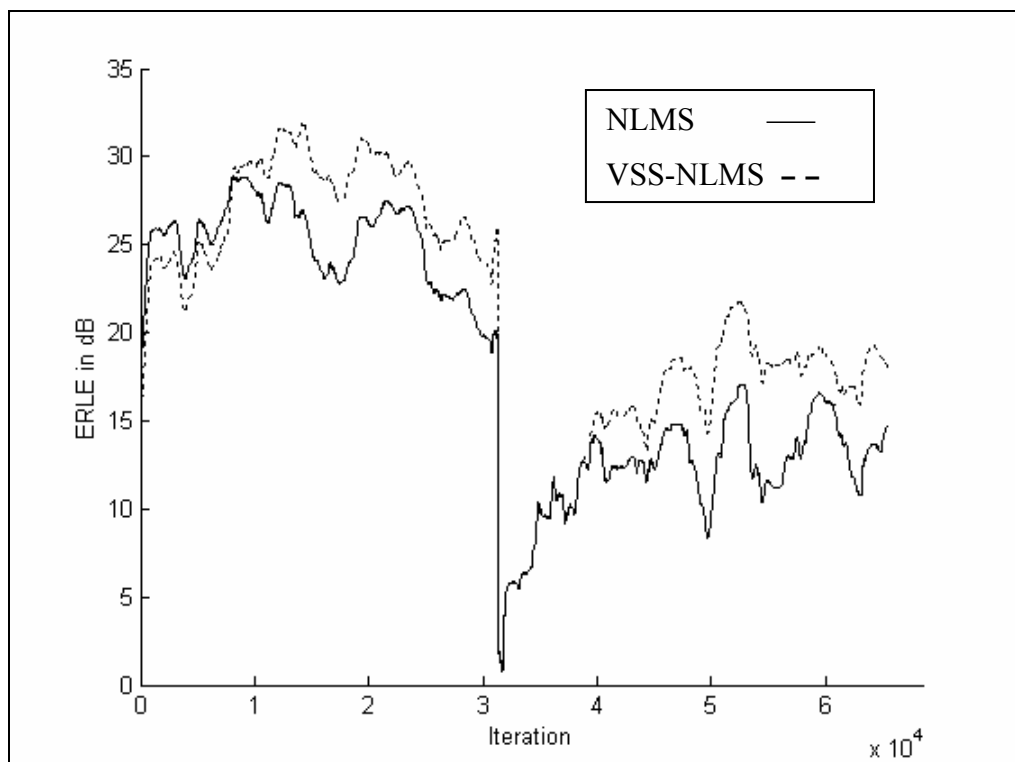


Figure 5.8. ERLE curves for NLMS and VSS-NLMS algorithms in 32 ms AEC

The simulation results show that the VSS and the GAS NLMS algorithms have the same converge speed of the NLMS algorithm. When the MSD curves are examined, they may have slightly slower speed due to the fluctuations in the step size update. This situation can be observed for the VSS-NLMS algorithm in Figure 5.7 and Figure 5.8. The GASS- $\alpha$ -S-NLMS algorithm also converges slightly slower especially after instantaneous change due to the fluctuations in the step size update. However, when the residual echoes are listened for subjective tests, the echo suppression speed has the same quality for all of the algorithms in 32 ms AEC simulation. For longer filter lengths, the GAS methods convergence slower especially after the instantaneous change. Due to this reason and also considering its high computational complexity, we would not use the GAS-APA in the 256 ms AEC simulations. The ERLE values demonstrate the steady state performance superiority of the VSS and the GAS algorithms.

In Table 5.5, we demonstrate the performance results of the subband and the fullband APA(2), the VSS-APA(2), the GASS- $\alpha$ -S-APA(2), the EW-APA(2) and the SEW-APA(2) in the 256 ms AEC simulation.

Table 5.5. Results of the 256 ms AEC simulation for the subband and the fullband APA(2), the VSS-APA(2), the GASS- $\alpha$ -S-APA(2), the EW-APA(2) and the SEW-APA(2)

Algorithm	$n$ from 1 to inst. change			After inst. change		
	ERLE (dB)	CS (sec)	$E[\mu(n)]$	ERLE (dB)	CS (sec)	$E[\mu(n)]$
Subband APA(2)	32	3	1	23	2.3	1
APA(2)	28	3	1	18	2.3	1
VSS-APA(2)	34	3	0.1	23	2.3	0.2
EW-APA(2)	31	4	1	22	3.8	1
SEW-APA(2)	32	3.5	1	23	2.5	1

The VSS-APA(2) has the convergence speed of the APA(2), and has the best steady state performance. The SEW-APA(2) also provide fast convergence and large ERLE. However, the EW-APA(2) provide slow convergence especially with the instantaneous change. The subband adaptive filter also provides both fast convergence and large ERLE value with lower computational complexity than that of the fullband [2]. However, the delay introduced by the analysis and the synthesis filter banks, (here nearly 12 ms), is the main drawback of the subband filtering.

#### 5.4. Inverse System Identification

In the inverse system identification problem, the output signal of the unknown system is applied as the input of the adaptive filter. The adaptive filtering algorithm adjusts the parameters of the filter in order to model the inverse response of the unknown system. The inverse system identification setup is shown in Figure 5.9.

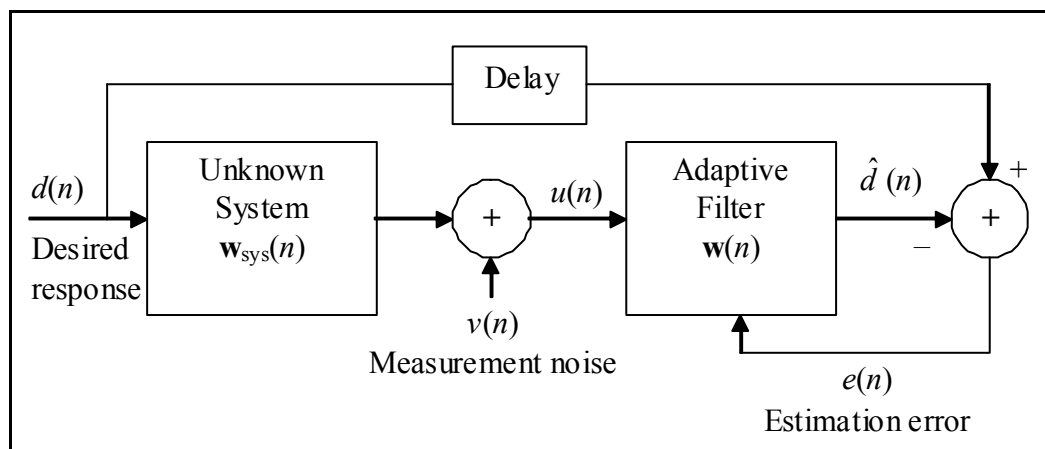


Figure 5.9. Inverse system identification configuration

The unknown system with impulse response  $\mathbf{w}_{\text{sys}}(n)$  may have more than one inverse impulse response (depending on number of zeros in transfer function  $\mathbf{W}_{\text{sys}}(z)$ ). However, only one of the inverse responses is stable. The stable inverse impulse response,  $\mathbf{w}_{\text{inv}}(n)$ , is causal, only when the impulse response  $\mathbf{w}_{\text{sys}}(n)$  is minimum phase that is  $\mathbf{W}_{\text{sys}}(z)$  has its zeros and poles in the unit circle and the stable inverse has a region of convergence that goes to infinity. When one zero of the  $\mathbf{W}_{\text{sys}}(z)$  is outside the unit circle (non-minimum phase system), the stable inverse will have a region of convergence between innermost and

outermost zeros closest to the unit circle (zeros of  $\mathbf{W}_{sys}(z)$  are the poles of inverse  $\mathbf{W}_{sys}(z)$ ,  $\mathbf{W}_{inv}(z)$ ). This inverse response  $\mathbf{w}_{inv}(n)$  is a two-sided, noncausal sequence. To be able to obtain such noncausal inverse, a delay is introduced into the inverse system identification configuration. The noncausal inverse response starts from an iteration point and spreads to both sides of the axes. The starting iteration point is usually negative and is approximately equal to the negative of the group delay of the  $\mathbf{w}_{sys}(n)$ , when  $\mathbf{w}_{sys}(n)$  is a linear phase system. The delay introduced into inverse system identification aims to model the two-sided stable inverse by centering it in the middle of the adaptive filter. Therefore, any truncation on both sides due to the finiteness of the adaptive filter causes minimal estimation error. The delay term is equal to the sum of the two terms: half of the filter length and the group delay of the unknown system.

In this section, we will demonstrate the performance improvements provided by the VSS and the GASS NLMS algorithms in an inverse system identification simulation. The other proposed algorithms also show behaviors similar to that of in the system identification, hence they are not presented in this section. The random walk model is used as the nonstationary unknown system model. The noise vector  $\boldsymbol{\omega}(n)$  is a zero mean white Gaussian process vector with a diagonal autocorrelation matrix  $\mathbf{Q} = \sigma_{\omega}^2 \mathbf{I}$ . The variance  $\sigma_{\omega}^2$  is equal to  $10^{-7}$ , which means very slow variation. The variance values higher than this value cause too fast variations in the inverse response, hence tracking of the inverse impulse response becomes such that even step size of one is small to get satisfactory tracking.

In the nonstationary inverse system identification problem, the desired response  $d(n)$  is a white Gaussian process having zero mean and unit variance. The system impulse response  $\mathbf{w}_{sys}(n)$  is length of three, and it varies with the random walk model defined above. The initial system response is as follows:

$$\mathbf{w}_{sys}(0) = \begin{cases} \frac{1}{2} \left[ 1 + \cos\left(\frac{2\pi}{3.1}(m-2)\right) \right] & m = 1, 2, 3 \\ 0 & \text{otherwise} \end{cases} \quad (5.20)$$

The system in (5.20) has two zeros of  $-3.2679$  and  $-0.306$  (non-minimum phase) and it has a linear phase response (the group delay is one). An adaptive transversal filter of length 20 is used in the simulation. A delay of 11 (half filter length plus group delay,  $20/2+1$ ) is introduced to the desired response of the adaptive filter. The measurement noise (white Gaussian process with zero mean and variance  $\sigma_v^2$  of  $10^{-2}$ ) is also added to the output of the unknown system. At iteration  $20.0 \times 10^3$ , an instantaneous change to unknown system is applied by setting it equal to the negative of start up value ( $\mathbf{w}_{\text{sys}}(20.000) = -\mathbf{w}_{\text{sys}}(0)$ ). The measurement noise variance is also increased to  $4.0 \times 10^{-2}$ . The eigenvalue spread of  $20 \times 20$  autocorrelation matrix  $\mathbf{R}$  of the input process  $u(n)$  is approximately 11.5 ( $\lambda_{\text{max}} = 2.3$ ,  $\lambda_{\text{min}} = 0.2$ ). The simulation results are obtained by ensemble averaging 10 independent runs. Before proceeding to the results, the parameter values for the algorithms are given as:

- NLMS algorithm has  $\mu = 1$ .
- VSS-NLMS algorithm has;  $m = 2$ ,  $\gamma = 0.1$ ,  $\beta = 0.8$  and  $\theta = 0.999$ .
- GASS- $\alpha$ -S-NLMS algorithm has  $\alpha = 0.99$  and  $\rho = 0.04$ .
- The algorithms use a regularization parameter value of  $10^{-2}$  ( $\delta = 10^{-2}$ ).
- For the GAS and the VSS algorithms, the step size is bounded with  $\mu_{\text{max}} = 1$  and  $\mu_{\text{min}} = 10^{-3}$ .

In Table 5.6, the performances of the NLMS, the VSS-NLMS and the GASS- $\alpha$ -S-NLMS algorithms are demonstrated. In this simulation, the MSD curves are not available, hence convergence speeds are also calculated from the MSE curves.

The simulation results show that the performances of the VSS and the GASS- $\alpha$ -S-NLMS algorithms are very similar to that of the nonstationary system identification. Both of them have the convergence speed of the NLMS algorithm, but only the GAS can track the minimum misadjustment.

Table 5.6. Simulation results for the NLMS, the VSS-NLMS and the GASS- $\alpha$ -S-NLMS algorithms

Algorithm	$n$ from 1 to 19999			$n$ from 20000 to 40000		
	$M$ (%)	$CS$ (ite.)	$E[\mu(n)]$	$M$ (%)	$CS$ (ite.)	$E[\mu(n)]$
NLMS	430	150	1	308	150	1
VSS-NLMS	110	150	0.02	133	150	0.3
GASS- $\alpha$ -S-NLMS	120	150	0.008	60	150	0.016

## 6. DISCUSSIONS ON THE PROPOSED ALGORITHMS

In this section, the proposed algorithms will be discussed regarding their performances and computational complexities.

We have proposed the ASS-APA (GASS and GAMS), hence we obtain algorithms having both fast convergence and minimum misadjustment for the nonstationary filtering problems. The VSS-APA is also proposed, and it provides fast convergence and minimum misadjustment for the stationary filtering problems. Although, the VSS-APA can not provide the tracking performance of the GAS-APA in nonstationarity, it can provide better steady state performance in stationarity, especially when the signal to noise ratio is low. Therefore, when the optimum solution is stationary, the VSS-APA should be preferred over the GAS-APA.

We have also proposed the EW-APA and demonstrated its steady state performance superiority over the APA. But, the EW-APA has the convergence speed loss problem when the optimum solution changes instantaneously. It has convergence speed - misadjustment trade off problem, similar to the RLS algorithm. We demonstrated that GAS-EW-APA may partially remove the trade off problem in the nonstationary filtering problem. We also proposed the switching method (SEW-APA), hence both fast convergence of the APA and small misadjustment of the EW-APA are obtained. The SEW-APA may provide performances similar to the GAS-APA and the VSS-APA in nonstationarity and stationarity, respectively.

Beside the performances, the computational complexities of the proposed algorithms are also important. The simplified versions of the GAS-APA are more preferable than the GAS-APA, because they can provide the same performance with much less computational complexity. The GAMS methods have more computational burden than the GASS methods, but they are not superior to the GASS. Therefore, the GASS-APA is more preferable than the GAMS-APA. Among the proposed algorithms, the VSS-APA has the lowest computational complexity and it best suits the applications in which the optimum solution is stationary. The EW-APA replaces the matrix inversion operation in the APA

with the recursive estimation of the inverse correlation matrix, which has less computational complexity. However, the switching method requires calculation of the zero forcing solution in convergence, and also update of the matrix (that is used in the matrix inversion of the zero forcing solution) in the steady state. The estimation of the inverse correlation matrix may be frozen in the convergence, if this matrix (so input process) is nearly time invariant.

In Table 6.1, we demonstrate the computational complexities of the proposed methods, where  $M$  is the filter length and  $P$  is the projection order. For the VSS and the GASS- $\alpha$ -S methods, we demonstrate the extra computations added to the APA. For the switching method, the extra computations added to the EW-APA are given.

Table 6.1. Computational complexities of the proposed methods

Method	Computational Operations
APA(P)	$P \times P$ matrix inversion, $P^3 + (M+1)P^2 + M + P$ multiplications, $P^3 + MP^2 - MP + 2M - 1$ additions
EW-APA(P)	$5P^2 + P$ multiplications, 1 division $3P^2 - P$ additions
VSS added APA(P)	2 additions, 6 multiplications
GASS- $\alpha$ -S added APA(P)	$2M + 1$ additions, $M + 2$ multiplications
Switching added EW- APA(P)	$2P + 1$ additions, $P + 3$ multiplications

## 7. CONCLUSIONS

In the thesis, we proposed algorithms that provide both fast convergence and small misadjustment. The Affine Projection Algorithm is chosen regarding its convergence character. The Affine Projection Algorithm is a member of the Zero Forcing Algorithms; hence it converges fast due to the zero forcing. But, it also inherits the noise amplification drawback of the zero forcing, which causes large misadjustment. To obtain both fast convergence and minimum misadjustment, the adaptive step size methods are proposed for the APA. The Variable Step Size APA is proposed especially for the filtering problems with the time invariant optimum solution. The Gradient Adaptive Step size APA is proposed for the filtering problems in a nonstationarity environment. The Exponentially Weighted APA and its switching version (SEW-APA) are also proposed and shown to have similar performances to the VSS-APA in stationary and the GAS-APA in nonstationary cases, respectively.

The Adaptive Step Size methods proposed for the Affine Projection Algorithm are the Variable Step Size and the Gradient Adaptive Step size. The fast convergence and minimum misadjustment provided by the algorithms are demonstrated with system identification, echo cancellation and inverse system identification simulations. Furthermore, the Exponentially Weighted APA is proposed and it is demonstrated that it also has the convergence speed – misadjustment trade off problem similar to that of the RLS algorithm. The Switching Exponentially Weighted APA (SEW-APA) is proposed to remove that trade off problem; hence performance similar to the ASS-APA is obtained. It is shown that the SEW-APA can be a computationally simpler alternative of the GAS-APA.

### 7.1. Suggestions for Future Work

The Adaptive Step size methods proposed in the thesis may be applied to the Fast Affine Projection and the Block Exact Fast Affine Projection [23, 24] algorithms to obtain new algorithms providing fast convergence and minimum misadjustment with less computational complexity than that of the ASS-APA.

## APPENDIX A: GRADIENT-BASED ALGORITHMS

The Gradient-Based Algorithms are the adaptive linear filtering algorithms, which use gradient-based adaptation to obtain the optimum filter parameters (optimum Wiener solution) [1,2]. In gradient-based adaptation, the gradient of the performance function (usually mean squared error) with respect to the adaptive filter's tap-weight vector is used to update the adaptive filter's tap-weight vector.

Gradient-based adaptation originates from an optimization technique known as the method of steepest descent. The fundamental gradient-based algorithm, which is derived by using the method of steepest descent and Wiener filtering theory, is known as Steepest Descent Algorithm [1].

### A.1. Steepest Descent Algorithm

Before introducing the Steepest Descent Algorithm (SDA), the Wiener filter theory has to be stated, since this class of linear optimum filtering establishes the mathematical foundations of gradient-based algorithms. The purpose of the linear filtering methods is to reach a solution, which is optimal in a sense. The optimum Wiener solution for the linear filtering problem is obtained by using a statistical optimization method, which involves minimization of the cost function named Mean Squared Error (MSE):

$$J(n) = E[|e(n)|^2] \tag{A.1}$$

where  $E[.]$  is the expectation operator and  $e(n)$  is the estimation error defined as the difference between the desired response  $d(n)$  and the linear filter output  $y(n)$ .

The input process  $u(n)$  and the tap-weight coefficients  $w_k(n)$  can be defined in vector form as follows:

$$\mathbf{u}(n) = [u(n) \ u(n-1) \ \dots \ u(n-M+1)]^T \tag{A.2}$$

$$\mathbf{w}(n) = [w_0(n) \ w_1(n) \ \dots \ w_{M-1}(n)]^T \quad (\text{A.3})$$

The optimum Wiener solution is stated by the Wiener-Hopf equation as:

$$\mathbf{R} \mathbf{w}_{opt} = \mathbf{p} \quad (\text{A.4})$$

where  $\mathbf{w}_{opt}$  is the optimum Wiener solution that makes the cost function  $J(n)$  minimum.  $\mathbf{R}$  and  $\mathbf{p}$  are the autocorrelation matrix of the input and the cross correlation vector between the desired response and the input, respectively. They are defined as follows:

$$\mathbf{R} = E[\mathbf{u}(n)\mathbf{u}^H(n)] \quad (\text{A.5})$$

$$\mathbf{p} = E[\mathbf{u}(n)d^*(n)] \quad (\text{A.6})$$

The cost function can be expanded as follows:

$$\begin{aligned} J(n) &= E[|e(n)|^2] \\ &= E[|d(n)|^2] - \mathbf{w}^H(n)\mathbf{p} - \mathbf{p}^H \mathbf{w}(n) + \mathbf{w}^H(n)\mathbf{R}\mathbf{w}(n) \end{aligned} \quad (\text{A.7})$$

To find the minimum value of the cost function,  $\mathbf{w}$  is replaced by the optimum Wiener solution value  $\mathbf{w}_{opt}$  ( $\mathbf{w}_{opt} = \mathbf{R}^{-1}\mathbf{p}$ ) and the minimum error is found as:

$$J_{min} = E[|d(n)|^2] - \mathbf{p}^H \mathbf{R}^{-1}\mathbf{p} \quad (\text{A.8})$$

Then, the cost function equation becomes:

$$J(n) = J_{min} + (\mathbf{w}(n) - \mathbf{w}_{opt})^H \mathbf{R} (\mathbf{w}(n) - \mathbf{w}_{opt}) \quad (\text{A.9})$$

The Steepest Descent Algorithm is a gradient-based tap-weight adaptation method. The gradient of the cost function with respect to the tap-weight vector is used as the update term for the tap-weight vector. Since the cost function  $J(n)$  is a quadratic function of the tap-weights, negative of the gradient vector points toward the optimum tap-weight at which the cost function is minimum. The tap-weight adaptation is defined as:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \frac{1}{2} \mu [-\nabla J(n)] \quad (\text{A.10})$$

where  $\mu$  is the step size parameter, and the gradient is:

$$\begin{aligned} \nabla J(n) &= -2 [\mathbf{p} - \mathbf{R}\mathbf{w}(n)] \\ &= -2 E[e^*(n)\mathbf{u}(n)] \end{aligned} \quad (\text{A.11})$$

When (A.11) is substituted into (A.10), the tap-weight update rule for the SDA is obtained as follows:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu E[e^*(n)\mathbf{u}(n)] \quad (\text{A.12})$$

The mean of the tap-weight vector reaches to the optimum Wiener solution in the steady state.

## A.2. Least Mean Square Algorithm

The SDA as a gradient-based algorithm uses a deterministic gradient in the tap-weight update. There is another class of gradient-based algorithms, which is known as Stochastic Gradient Algorithms. The Least Mean Square (LMS) algorithm is an important member of this family.

The LMS algorithm uses the simplest estimators for  $\mathbf{R}$  and  $\mathbf{p}$  in the estimation of the gradient vector [1]; these are instantaneous values:

$$\mathbf{R} \cong \mathbf{u}(n)\mathbf{u}^H(n) \quad (\text{A.13})$$

$$\mathbf{p} \cong \mathbf{u}(n)d^*(n) \quad (\text{A.14})$$

$$\nabla J(n) = -2 [\mathbf{u}(n)d^*(n) - \mathbf{u}(n)\mathbf{u}^H(n)\mathbf{w}(n)] \quad (\text{A.15})$$

Using the instantaneous gradient estimate, the tap-weight update rule for the LMS algorithm is as follows:

$$\begin{aligned}\mathbf{w}(n+1) &= \mathbf{w}(n) + \mu \mathbf{u}(n) [d^*(n) - \mathbf{u}^H(n) \mathbf{w}(n)] \\ &= \mathbf{w}(n) + \mu \mathbf{u}(n) e^*(n)\end{aligned}\tag{A.16}$$

The mean of the tap-weight vector of the LMS algorithm also reaches to the optimum Wiener solution in the steady state.

### A.2.1. Performance Analysis of LMS algorithm

The analysis of the LMS algorithm is based on the mean squared error cost function. In the analysis, the weight error vector is used:

$$\boldsymbol{\epsilon}(n) = \mathbf{w}(n) - \mathbf{w}_{opt}\tag{A.17}$$

where  $\mathbf{w}_{opt}$  is the optimum Wiener solution.

Subtracting  $\mathbf{w}_{opt}$  from both sides of the tap-weight update (A.16), and using (A.17), we get the update equation for the weight error vector as:

$$\boldsymbol{\epsilon}(n+1) = [\mathbf{I} - \mu \mathbf{u}(n) \mathbf{u}^H(n)] \boldsymbol{\epsilon}(n) + \mu \mathbf{u}(n) v^*(n)\tag{A.18}$$

where  $\mathbf{I}$  is the identity matrix and  $v(n)$  is the measurement noise defined as the estimation error produced in the optimum Wiener solution:

$$v(n) = d(n) - \mathbf{w}_{opt}^H \mathbf{u}(n)\tag{A.19}$$

Before proceeding to the analysis, we will state some assumptions required for the mathematical tractability of the analysis. These assumptions are referred as the independence assumptions [1]:

- Tap input vectors  $\mathbf{u}(1)$ ,  $\mathbf{u}(2)$ , ...  $\mathbf{u}(n)$  are statistically independent vectors.
- $\mathbf{u}(n)$  is statistically independent of all previous samples of the desired response, namely,  $d(1)$ ,  $d(2)$ , ...,  $d(n-1)$ .

- $d(n)$  is dependent on  $\mathbf{u}(n)$ , but statistically independent of all previous samples of the desired response.
- The input vector  $\mathbf{u}(n)$  and the measurement noise  $v(n)$  are independent of each other.
- The measurement noise  $v(n)$  is white with zero mean and finite variance,  $\sigma_v^2 < \infty$ .

The correlation matrix of the weight error vector  $\boldsymbol{\epsilon}(n)$  is defined as:

$$\mathbf{K}(n) = E[\boldsymbol{\epsilon}(n) \boldsymbol{\epsilon}^H(n)] \quad (\text{A.20})$$

Using this definition, (A.18) and the independence theory, we get

$$\mathbf{K}(n+1) = (\mathbf{I} - \mu\mathbf{R})\mathbf{K}(n)(\mathbf{I} - \mu\mathbf{R}) + \mu^2\sigma_v^2\mathbf{R} \quad (\text{A.21})$$

To obtain a steady state solution of the difference equation (A.21),  $n$  is large, we may set  $\mathbf{K}(n+1) = \mathbf{K}(n)$ . Using the assumption that step size parameter  $\mu$  is small enough to ignore the term  $\mu^2\mathbf{R}\mathbf{K}(n)\mathbf{R}$  in comparison with the identity matrix  $\mathbf{I}$ . Then we may approximate the (A.21) as follows:

$$\mathbf{R}\mathbf{K}(n) + \mathbf{K}(n)\mathbf{R} \cong \mu\sigma_v^2\mathbf{R} \quad (\text{A.22})$$

Here, we will introduce a new performance criterion to assess the performance of an adaptive filter; Mean Square Deviation (MSD) which is defined as:

$$\begin{aligned} D(n) &= E[\|\mathbf{w}(n) - \mathbf{w}_{opt}(n)\|^2] \\ &= E[\|\boldsymbol{\epsilon}(n)\|^2] \end{aligned} \quad (\text{A.23})$$

Equation (A.23) can be reformulated as

$$D(n) = \text{tr}[\mathbf{K}(n)] \quad (\text{A.24})$$

where  $\text{tr}[\cdot]$  is the trace of a matrix (sum of main diagonal elements).

When we multiply both sides of (A.22) by the inverse matrix  $\mathbf{R}^{-1}$  and take the trace of both sides, the mean square deviation of the LMS algorithm is obtained as:

$$D(n) = \frac{1}{2} \mu M \sigma_v^2 \quad (\text{A.25})$$

where  $M$  is the adaptive filter length.

The estimation error could be expressed in terms of  $v(n)$  and  $\mathbf{\epsilon}(n)$  by using (A.17) and (A.19) as follows:

$$e(n) = v(n) - \mathbf{\epsilon}^H(n) \mathbf{u}(n) \quad (\text{A.26})$$

Using (A.26), the performance index of the LMS algorithm, mean square error cost function, may be redefined as

$$J(n) = J_{\min} + E[\mathbf{\epsilon}^H(n) \mathbf{u}(n) \mathbf{u}^H(n) \mathbf{\epsilon}(n)] \quad (\text{A.27})$$

where  $J_{\min}$  could be shown to be equal to  $\sigma_v^2$ . Taking trace of both sides and invoking the independence assumption, we have

$$J(n) = J_{\min} + \text{tr}[\mathbf{R}\mathbf{K}(n)] \quad (\text{A.28})$$

The second term of addition in (A.28) is defined as the excess mean squared error:

$$J_{exc}(n) = \text{tr}[\mathbf{R}\mathbf{K}(n)] \quad (\text{A.29})$$

Taking the trace of both sides in (A.22), we could approximate the excess MSE:

$$J_{exc}(\infty) = \frac{1}{2} \mu \sigma_v^2 \text{tr}[\mathbf{R}] \quad (\text{A.30})$$

In the steady state, the ratio of the excess error  $J_{exc}(\infty)$  to the minimum error  $J_{\min}$  is called the misadjustment  $M$  and it is stated as:

$$\begin{aligned}
M &= \frac{J_{exc}(\infty)}{J_{\min}} \\
&= \frac{1}{2} \mu \operatorname{tr}[\mathbf{R}]
\end{aligned}
\tag{A.31}$$

The misadjustment is used as a performance index and it tells us how close the excess error to the minimum error is. This ratio is usually expressed in percentage.

The cost function  $J(n)$  of the LMS algorithm is expected to converge in the mean. The convergence requirement imposes a bound on the design parameter; step size. This bound could be established as follows:

The correlation matrix  $\mathbf{R}$  could be transformed by a unitary similarity transformation:

$$\mathbf{Q}^H \mathbf{R} \mathbf{Q} = \mathbf{\Lambda} \tag{A.32}$$

where  $\mathbf{\Lambda}$  is a diagonal matrix consisting of the eigenvalues of the correlation matrix  $\mathbf{R}$ ,  $\mathbf{Q}$  is the unitary matrix having eigenvectors associated with these eigenvalues as columns. Using this transformation,  $J_{exc}(n)$  in (A.29) could be rewritten as:

$$\begin{aligned}
J_{exc}(n) &= \operatorname{tr}[\mathbf{R} \mathbf{K}(n)] \\
&= \operatorname{tr}[\mathbf{\Lambda} \mathbf{X}(n)]
\end{aligned}
\tag{A.33}$$

where  $\mathbf{X}(n)$  is the transformed weight error correlation matrix (transform  $\mathbf{Q}^H$  used) and defined as

$$\mathbf{X}(n) = \mathbf{Q}^H \mathbf{K}(n) \mathbf{Q} \tag{A.34}$$

Since  $\mathbf{\Lambda}$  is a diagonal matrix, we may also write

$$J_{exc}(n) = \sum_{i=1}^M \lambda_i x_i(n) \tag{A.35}$$

where the  $x_i(n)$  is the  $i^{\text{th}}$  diagonal element of the matrix  $\mathbf{X}(n)$ .

Using transformations described by (A.32) and (A.34), we may rewrite the recursive equation (A.21) in terms of  $\mathbf{X}(n)$  and  $\mathbf{\Lambda}$  as follows:

$$\mathbf{X}(n+1) = (\mathbf{I} - \mu\mathbf{\Lambda})\mathbf{X}(n)(\mathbf{I} - \mu\mathbf{\Lambda}) + \mu^2\sigma_v^2\mathbf{\Lambda} \quad (\text{A.36})$$

From (A.35), we see that  $J_{exc}(n)$  depends on the  $x_i(n)$ , diagonal elements, and then we have

$$x_i(n+1) = (1 - \mu\lambda_i)^2 x_i(n) + \mu^2\sigma_v^2\lambda_i \quad i = 1, 2, \dots, M \quad (\text{A.37})$$

$x_i(n)$  has to be convergent in the mean, and then  $J_{exc}(n)$  converges too. This requirement brings the stability bound on the step size as:

$$0 < \mu_i < \frac{2}{\lambda_i} \quad i = 1, 2, \dots, M \quad (\text{A.38})$$

where  $\mu_i$  is the step size associated with the  $i^{\text{th}}$  eigenvalue. This assumes the step size as a diagonal matrix having diagonal elements  $\mu_i$ . From (A.37), one can show that the fastest convergence can be obtained when

$$\mu_i = \frac{1}{\lambda_i} \quad (\text{A.39})$$

This means, using the eigenvalue reciprocals as a diagonal step size matrix in the LMS algorithm could provide the fastest convergence. This is only valid when the LMS algorithm operates in the transform domain [1], where the input correlation and the weight error correlation matrices are diagonalized by the unitary transform  $\mathbf{Q}^H$ .

When a scalar step size is employed, the step size must be bounded for stability as

$$0 < \mu < \frac{2}{\lambda_{\max}} \quad (\text{A.40})$$

where  $\lambda_{\max}$  is the maximum eigenvalue of  $\mathbf{R}$ . This bound is valid in both transform and time domains. Using a constant step size in both domains gives the same cost function  $J(n)$  (as (A.33) states).

Equation (A.39) also states that when a constant step size  $\mu$  is used,  $\lambda_{\min}$ , which is the minimum eigenvalue of  $\mathbf{R}$ , determines the convergence speed since it constitutes the slowest mode in the transient state. Assume that the step size value is attained as

$$\mu = \alpha \frac{2}{\lambda_{\max}} \quad (\text{A.41})$$

where  $\alpha$  is a positive constant less than one ( $0 < \alpha < 1$ ). The slowest mode, determining convergence speed of the algorithm, is defined by the maximum value of the exponential decaying constant in (A.37) that is

$$\max[(1 - \mu\lambda_i)^2] = \left(1 - 2\alpha \frac{\lambda_{\min}}{\lambda_{\max}}\right)^2 \quad (\text{A.42})$$

where  $\max[.]$  means maximum value in the set. From (A.42), we observe that the convergence speed of the LMS algorithm is dependent on the ratio of the maximum and minimum eigenvalues of  $\mathbf{R}$ , and this ratio is known as the eigenvalue spread of  $\mathbf{R}$ ;  $\chi(\mathbf{R})$

$$\chi(\mathbf{R}) = \frac{\lambda_{\max}}{\lambda_{\min}} \quad (\text{A.43})$$

The LMS algorithm has a convergence speed characteristic, which is highly dependent on the eigenvalue spread of the input process. A white (uncorrelated), zero mean input process has a diagonal correlation matrix:

$$\mathbf{R} = \sigma_u^2 \mathbf{I} \quad (\text{A.44})$$

where  $\sigma_u^2$  is the variance of input process  $u(n)$  and  $\mathbf{I}$  is the identity matrix. The eigenvalue spread is equal to one, since all eigenvalues are equal to  $\sigma_u^2$ . But, a correlated input process

will have a non-diagonal correlation matrix, and its eigenvalue spread is higher than one. The LMS algorithm exhibits slower convergence, when the input process is more correlated and has higher eigenvalue spread.

To improve the convergence speed of the LMS algorithm with correlated input processes, a method is needed to remove the effect of the high eigenvalue spread of the correlation matrix. Firstly, we have to understand how the correlation matrix affects the tap-weight update in the LMS algorithm. Using tap-weight update (A.16), (A.17) and (A.26), one can obtain:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \mathbf{u}(n) [\mathbf{u}^H(n) (\mathbf{w}_{opt}(n) - \mathbf{w}(n)) + v^*(n)] \quad (\text{A.45})$$

The tap-weight update term is wanted to be equal to the weight error vector ( $\mathbf{w}_{opt} - \mathbf{w}(n)$ ). Then,  $\mathbf{w}(n+1)$  becomes equal to  $\mathbf{w}_{opt}$ . However, the weight error vector exists as multiplied with the term  $\mu \mathbf{u}(n) \mathbf{u}^H(n)$  (step size and instantaneous correlation matrix). This means it is transformed by the instantaneous correlation matrix and multiplied with the step size. In addition, the measurement noise  $v(n)$  term also exists as a distortion. As iterations proceed, the tap-weight update makes an averaging over the instantaneous correlation matrix and then the weight error vector becomes as if it is transformed by the correlation matrix. The convergence of the tap-weight vector is favored in some eigenvector directions of the correlation matrix  $\mathbf{R}$  (in the directions having large eigenvalues) and it is slowed down in the eigenvector directions having small eigenvalues. Therefore, the eigenvectors having small eigenvalues determine the convergence speed.

In this case, one way to remove the effect of  $\mathbf{R}$  on the convergence is to use a tap-weight update of the form:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \hat{\mathbf{R}}^{-1} \mathbf{u}(n) e^*(n) \quad (\text{A.46})$$

where  $\hat{\mathbf{R}}^{-1}$  is the estimate of the inverse correlation matrix. Using this method, the rotation of the correlation matrix is removed by its inverse; hence fast convergence can be obtained regardless of the eigenvalue spread of the correlation matrix. Equation (A.46) defines the

Newton (or LMS- Newton) Algorithm [1]. Zero forcing form of the Newton algorithm is known as Quasi-Newton algorithm. This form uses a zero forcing scalar step size [20].

### A.3. Transform Domain LMS Algorithm

As introduced in the previous section, the conventional LMS algorithm converges slowly with highly correlated input signals. The use of estimate of the inverse correlation matrix may be impractical due to its high computational burden.

Another method to increase the convergence speed of the LMS algorithm is to operate it in the transform domain where the eigenvalue reciprocals can be used as the step size values. Using the unitary matrix  $\mathbf{Q}$  introduced in (A.32) to transform the input vector  $\mathbf{u}(n)$ , we switch into the transform domain:

$$\mathbf{v}(n) = \mathbf{Q}^H \mathbf{u}(n) \quad (\text{A.47})$$

where  $\mathbf{Q}$  is the unitary matrix having eigenvectors of  $\mathbf{R}$  as columns. The correlation matrix of the transformed  $\mathbf{u}(n)$  is a diagonal matrix:

$$E[\mathbf{v}(n)\mathbf{v}^H(n)] = \mathbf{\Lambda} \quad (\text{A.48})$$

which has the eigenvalues of the correlation matrix  $\mathbf{R}$  as its diagonal elements.

Transformation in (A.47) is known as Karhunen-Loéve Transformation (KLT) [1]. This transformation requires estimation of the eigenvectors that is signal dependent and computationally complex. Although we obtained a new diagonal correlation matrix in the transform domain, it still has the same eigenvalues and the eigenvalue spread with the correlation matrix  $\mathbf{R}$ . Therefore, using only  $\mathbf{v}(n)$  instead of  $\mathbf{u}(n)$  in the LMS algorithm does not improve the convergence speed. We need to use the inverse of the  $\mathbf{\Lambda}$  as a diagonal step size matrix. Hence, the eigenvalue reciprocals will be used as the step size values (This idea is stated in the previous section). We could state the Transform Domain LMS algorithm as follows:

$$e(n) = d(n) - \mathbf{h}^H(n)\mathbf{v}(n) \quad (\text{A.49})$$

$$\mathbf{h}(n+1) = \mathbf{h}(n) + \mu \Lambda^{-1}\mathbf{v}(n)e^*(n) \quad (\text{A.50})$$

where  $\mu$  is the scalar step size and

$$\Lambda^{-1} = \text{diag}(\lambda_1^{-1}, \lambda_2^{-1}, \dots, \lambda_M^{-1}) \quad (\text{A.51})$$

The optimum solution in the transform domain is equal to the transformed form of the time domain optimum solution:

$$\mathbf{h}_{opt} = \mathbf{Q}^H \mathbf{w}_{opt} \quad (\text{A.52})$$

TDLMS algorithm ideally requires KLT. KLT is a signal dependent and computationally complex transformation. Therefore, fixed orthogonal transforms, such as the Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), Discrete Sine Transform (DST), Walsh Hadamard Transform (WHT), Discrete Hartley Transform (DHT), Discrete Wavelet Transform (DWT) are used to approximate the KLT in adaptive filtering applications.

The TDLMS algorithm requires the estimation of reciprocal of eigenvalues in the  $\Lambda^{-1}$  matrix. The eigenvalues can be expressed as [1]:

$$\lambda_i = E[|v_i(n)|^2] \quad (\text{A.53})$$

where  $v_i(n)$  is the transformed input process at the  $i^{\text{th}}$  tap of the filter. The eigenvalues can be estimated by the instantaneous data as follows:

$$\hat{\lambda}_i(n) = \theta \hat{\lambda}_i(n-1) + (1-\theta) |v_i(n)|^2 \quad (\text{A.54})$$

where  $\theta$  is a positive scalar, close to one.

## REFERENCES

1. Haykin, S., *Adaptive Filter Theory*, Prentice-Hall, Upper Saddle River, 2000.
2. Farhang-Boroujeny, B., *Adaptive Filters: Theory and Applications*, John Wiley & Sons, New York, 1998.
3. Benveniste, A., M. Metivier, and P. Priouret, *Adaptive Algorithms and Stochastic Approximation*, Springer-Verlag, New York, 1987.
4. Kushner, H. J. and J. Yang, "Analysis of adaptive step size SA algorithms for parameter tracking", *IEEE Transactions on Automatic Control*, Vol. 40, No. 8, pp. 1403-1410, August 1995.
5. Ang, W. P. and B. Farhang-Boroujeny, "A new class of gradient adaptive step size LMS algorithms", *IEEE Transactions on Signal Processing*, Vol. 49, No. 4, pp. 805-810, April 2001.
6. Farhang-Boroujeny, B., "Variable-step size LMS algorithm: new developments and experiments", *IEE Proceedings Visual Image Signal Processing*, Vol. 141, No. 5, pp. 311-317, October 1994.
7. Mathews, V. J. and Z. Xie, "A stochastic gradient adaptive filter with gradient adaptive step size", *IEEE Transactions on Signal Processing*, Vol. 41, No. 6, pp. 2075-2087, June 1993.
8. Woo, H. C., "Improved stochastic gradient adaptive filter with gradient adaptive step size", *Electronics Letters*, Vol. 34, No. 13, pp. 1300-1301, June 1998.
9. Harris, R. W., D. M. Chabries and F. A. Bishop, "A variable step (VS) adaptive filter algorithm", *IEEE Transactions on Acoustics Speech and Signal Processing*, Vol. ASSP-34, No. 2, pp. 309-316, April 1986.

10. Evans, J. B., P. Xue and B. Liu, "Analysis and implementation of variable step size adaptive algorithms", *IEEE Transactions on Signal Processing*, Vol. 41, No. 8, pp. 2517-2535, August 1993.
11. Gan, W. S., "Fuzzy step size adjustment for the LMS algorithm", *Signal Processing*, Vol. 49 pp. 145-149, 1996.
12. Kwong, R. H. and E. W. Johnston, "A variable step size LMS algorithm", *IEEE Transactions on Signal Processing*, Vol. 40, No. 7, pp. 1633-1642, July 1992.
13. Aboulnasr, T. and K. Mayyas, "A robust variable step size LMS-type algorithm: analysis and simulations", *IEEE Transactions on Signal Processing*, Vol. 45, No. 3, pp. 631-639, March 1997.
14. Jun, B. E. and D. J. Park, "Novel steepest descent adaptive filter derived from new performance function with additional exponential term", *Signal Processing*, Vol. 36, pp. 189-199, 1994.
15. Shan, T. J. and T. Kailath, "Adaptive algorithms with an automatic gain control feature", *IEEE Transactions on Circuits and Systems*, Vol. 35, No. 1, January 1988.
16. Song, S., J. S. Lim, S. J. Baek and K. M. Sung, "Gauss Newton variable forgetting factor recursive least squares for time varying parameter tracking", *Electronics Letters*, Vol. 36, No. 11, pp. 988-990, May 2000.
17. Sankaran, S. G. and A. A. (Louis) Beex, "Convergence behavior of Affine Projection Algorithms", *IEEE Transactions on Signal Processing*, Vol. 48, No. 4, pp. 1086-1096, April 2000.
18. Slock, D. T. M., "On the convergence behavior of the LMS and the normalized LMS algorithms", *IEEE Transactions on Signal Processing*, Vol. 41, No. 9, pp. 2811-2825, September 1993.

19. Diniz, P. S. R., M. L. R. Campos and A. Antoniou, "Analysis of LMS-Newton adaptive filtering algorithms with variable convergence factor", *IEEE Transactions on Signal Processing*, Vol. 43, No. 3, pp. 617-627, March 1995.
20. Diniz, P. S. R. and L. W. P. Biscainho, "Optimal variable step size for the LMS/Newton algorithm with application to subband adaptive filtering", *IEEE Transactions on Signal Processing*, Vol. 40, No. 11, pp. 2825-2829, November 1992.
21. Mader, A., H. Puder and G. U. Schmidt, "Step size control for acoustic echo cancellation filters – an overview", *Signal Processing*, Vol. 80, pp. 1697-1719, 2000.
22. Breining, C. and T. Schertler, "Delay-free low cost step-gain estimation for adaptive filters in acoustic echo cancellation", *Signal Processing*, Vol. 80, pp. 1721-1731, 2000.
23. Gay, S. L. and S. Tavathia, "The fast affine projection algorithm", *Proceedings of International Conference on Acoustics, Speech, Signal Processing*, Detroit, MI, May 8-12 1995, pp. 3023-3026.
24. Tanaka, M., S. Makino and J. Kojima, "A block exact fast affine projection algorithm", *IEEE Transactions on Speech and Audio Processing*, Vol. 7, No. 1, pp. 79-86, Jan. 1999.