Performance Study of The Leaky Least Mean Square Adaptive Algorithm With Delayed Adjustments

By

Farouk Laichi, B.A.Sc.

A dissertation submitted to the
School of Graduate Studies and Research,
University of Ottawa,
in partial fulfillment of the requirements
for the degree of
Master of Applied Science

Ottawa-Carleton Institute for Electrical Engineering
Department of Electrical Engineering
Faculty of Engineering
University of Ottawa

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

ISBN 0-315-82507-3

Canada

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

I hereby declare that I am the sole author of this thesis.

I authorize the University of Ottawa to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Farouk Laichi

I further authorize the University of Ottawa to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Farouk Laichi

Abstract

Adaptive filters constitute an important part of signal processing. They are widely used in many applications where signal statistics are unknown to the users. One of the most popular adaptive methods is known as the least mean-squared (LMS) algorithm. It is well known for its hardware simplicity and its cost effectiveness. However, the LMS adaptive algorithm does not operate well in non-ideal environments. Leakage in the updated equation of the LMS algorithm was proven to overcome many such problems. This algorithm is known as the leaky LMS (LLMS). The LLMS is very robust to drifting phenomena and overflow in registers; problems that occur in the digital implementation of adaptive algorithms. In many applications, an inherent delay in the feedback error path is encountered. This delay has a definite effect on the performance of the LLMS algorithm. In this thesis, we study the performance of LLMS in the presence of delay (LDLMS). This algorithm is basically the LLMS with a delay incorporated in the coefficient updated equation. A new general stability bound is derived for the LDLMS, from which bounds of convergence of LMS, LLMS, and delayed LMS(DLMS) can be obtained. Stability bounds, convergence behavior, and excess mean squared error for this new algorithm are investigated. Theoretical results are first derived and later verified by simulations. It will be shown that introducing leakage in the DLMS algorithm gives a compromise performance. Finally, examples of applications of the LDLMS algorithm are provided.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my thesis supervisor Dr. T. Aboulnasr and co-supervisor Dr. W. Steenaart. Their continued guidance and innumerable suggestions throughout my research are greatly appreciated. This thesis would not have been accomplished without their combined help.

My utmost thanks go to my family for their patience and support during my years of study.

I would like to thank all my fellow graduate students and the digital communications and signal processing group for the feedback they were able to give me during my research.

Finally, I would like to thank the Department of electrical Engineering, Ottawa University and TRIO funding which made this research possible.

Contents

List of Figures

Acronyms and Definitions

LMS	Least Mean-Squared
LLMS	Leaky Least Mean-squared
DLMS	Delayed Least Mean-squared
LDLMS	Leaky Delayed Least Mean-Squared
NLMS	Normalized Least Mean-squared
NDLMS	Normalized Delayed Least Mean-Squared
FIR	Finite Impulse Response
IIR	Infinite Impulse Response
MSE	Mean Squared Error
FSE	Fractional Spaced Equalizer
RLS	Recursive Least Square
ADPCM	Adaptive Differential Pulse Coded Modulation

Nomenclature

J	The cost function associated with either least squares or least mean squares
n	Current time index
N	Vector size or filter order
$e(n)$	The error computed at time iteration n
$e_{opt}(n)$	the optimum error vector at time n
$C(n)$ or C_n	The N $\times 1$ vector filter length
$< . >$	Denotes averaging over a signal statistics
C_{opt}	Denotes $N \times 1$ optimal weight vector
$d(n)$ or d_n	Denotes a scalar desired data samples at time n
P	$N \times 1$ cross-correlation vector
H	The input autocorrelation matrix
R_o	Denotes rotated optimum tap-weight vector
μ	Denotes the leakage parameter
V	The orthonormal eigenvector matrix
V_i	The i^{th} vector of the matrix V
Λ	Diagonal matrix of the eigenvalues of H
β	Denotes the step size parameter
D	Represents the delay time index
λ_i	The i^{th} element of Λ
γ	In one context was used as a substitute for $1 - \beta\mu$
ϕ	In one context was used as a substitute for $\beta\mu$
ν_x	Represents the kurtosis of the input distribution process

M	Represents $N + \nu_x - 1$
X_n or $X(n)$	This represent an $N \times 1$ input data vector at n
$\epsilon(n)$	Represents tap-weight error vector
$W(n)$ or W_n	Rotated versions of the weight error vector $epsilon(n)$
U_n	Rotated versions of the input vector X_n or $X(n)$
u_n	A single element of U_n
σ_x^2	Represents the variance of $x(n)$
λ_{avg}	Represents the average value of the elements of Λ
λ_{rms}	Represents the effective root mean square of the elements of Λ
λ_{max}	Represents the maximum value of the elements of Λ
λ_{min}	Represents the minimum value of the elements of Λ
α	Represents the ratio of λ_{rms} over λ_{avg}
J_{min}	the minimum mean-squared error of e_n
$J_{ex}(n)$	Represents the excess mean-squared error at time n
$\mathbf{q}$	Represents the digital bias on the tap-weight vector
q	Denotes magnitude of $\mathbf{q}$
r	Denotes real root of a polynomial
$\mathcal{Z}$	Denotes the Z-tranform
z	Denotes the operator z of the Z-transform
δ_{km}	Denotes Kronecker delta function
$[.]^T$	Denotes transpose operation

Chapter 1

Introduction

Adaptive filters are an important part of signal processing. They are generally defined as filters whose characteristics can be modified to achieve desired objectives and accomplish this modification or adaptation automatically without user intervention. Due to the uncertainty about the input signal characteristics, the designer then turns to an adaptive filter which can learn the signal characteristics when first turned on and can later track changes in these characteristics. There is a continuously growing need for adaptive signal processing in areas such as prediction, system identification, beamforming, etc.

Many aspects of the adaptive filtering concepts are generally governed by the nature of the applications themselves. Adaptive techniques have generally been classified under two main categories. In one category, the cost function to be optimized in a running sum of squared errors is given by

$$J(n) = \min_{C(n)} \sum_{i=N-1}^{n-1} e_i^2(i), \quad N-1 \leq n \leq L-1, \tag{1.1}$$

where $e_i(n)$ denotes the error computed at time iteration i, N is the filter length, L is the length of the input data, and $C(n)$ is the tap-weight vector. This cost function uses all the available data from $i = N-1$ to $i = L-1$. Figure 1.1 shows a system identification model. This category of adaptive algorithms contains the whole class of Recursive Least Squares(RLS) algorithms, which includes Kalman algorithms, and Fast Transversal Filters algorithms [16], [17].

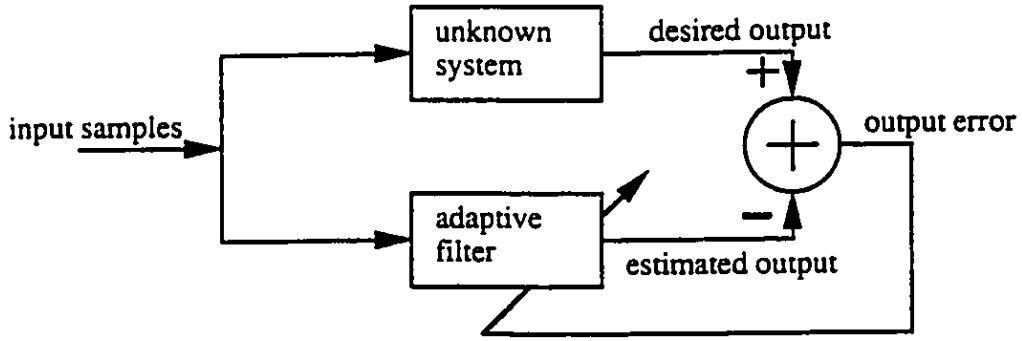
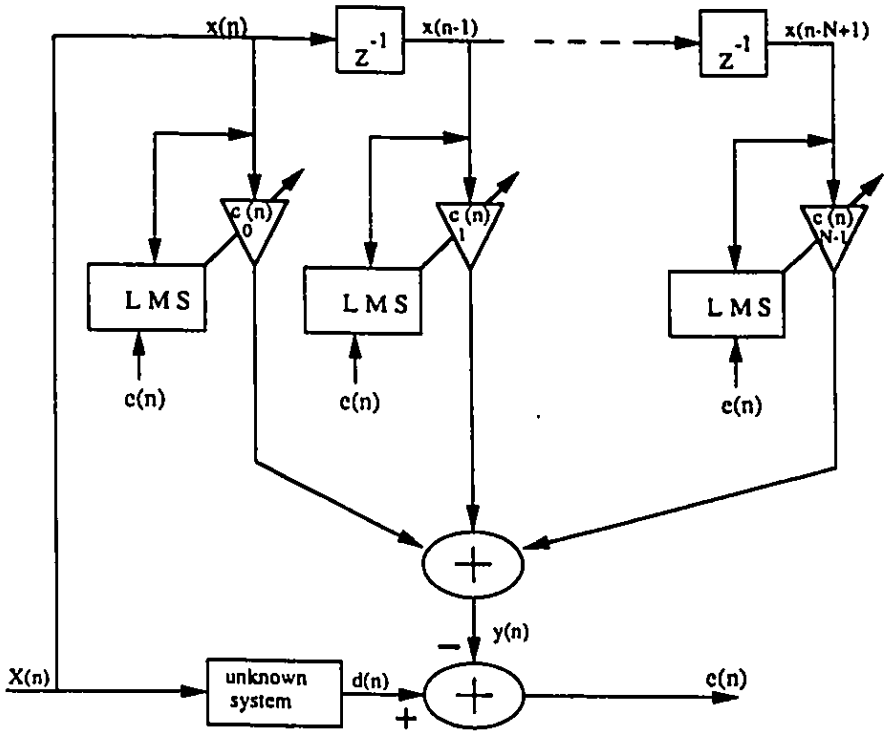


Figure 1 1: System identification model using an adaptive filter.

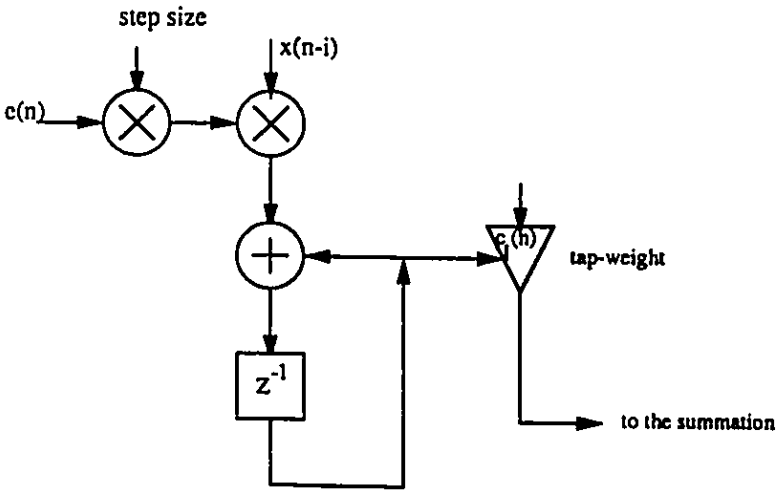
In the other category, the cost function to be optimized is a statistical measure of the squared error, known as the mean squared-error(MSE). This cost function is given by

$$J(n) = \min_{C(n)} < e^2(n) > \tag{1.2}$$

where $e(n)$ is the scaled error computed at time n and $< . >$ denotes statistical expectation. This category contains the whole class of gradient algorithms, which includes the least mean-squared (LMS) algorithm [18], [2], sign-LMS algorithm, normalized LMS(NLMS) algorithm, in addition to other recently developed adaptation schemes, all of which are LMS variants [2], [19], [20], [21]. The RLS class of algorithms offers faster convergence compared to the gradient-search type algorithms but usually at the cost of higher complexity and/or more numerical difficulties. In practice, they are useful in applications which need fast convergence. The LMS variants are relatively low in their computational requirements, work adequately in a variety of signal environments, and are the basis for many existing adaptive processors, such as adaptive equalizers and echo cancellers. Figure 1.2 shows the details of an FIR-based system identification model where the LMS algorithm is used to update the filter coefficients.



(a) system identification using the LMS algorithm



(b) update structure of the LMS algorithm

Figure 1.2: (a)System identification using the LMS algorithm. (b) LMS adaptive structure.

1.1 Problem Statement

In several applications of adaptive filters such as equalization of data transmission, system identification and adaptive differential pulse coded modulation (ADPCM), the convergence and stability of the stochastic gradient search algorithms such as LMS, can be problematic under non-ideal situations. Under such situations, the adaptive filter taps may drift away from their optimum settings and thus do not converge. In the following, we summarize the conditions under which adaptive algorithms may show unsatisfactory results [18], [22], [23].

1. *Insufficient spectral excitation*: The tap weights divergence of the adaptive algorithms are related to the inadequacy of excitation in the input process in its frequency bands.Therefore, due to this insufficient input energy, the adaptive filter will not be able to estimate the system response to these particular frequencies. Consequently, some or all of the filter taps will diverge or drift.

2. *Finite Precision Effects*: In a digital implementation, adaptive algorithm quantities are quantized to a certain finite precision. Quantization noise that results from this implementation may lead to deviations of the adaptive filter performance from what otherwise might have been expected. These deviations can be considerably greater than the least significant bit magnitude of the digital implemented quantities because of the continuous accumulation of the quantization errors with time. Then, register overflow occurs, and leads to an unacceptable filter performance.

3. *Stalling or biasing*: This is another condition where LMS-based algorithms are proven to not perform well. The stalling phenomenon happens because the gradient estimate is not sufficiently noisy to make the necessary adjustments to the adaptive tap-weights. Hence, the adaptation process stops. This stalling or lock up situation may arise in imperfect implementations. In a such case, stalling is not desirable as it biases the tap-weight filter possibly leading to longer convergence. A method to prevent stalling phenomenon is to insert dither at the input of the

quantizer that feeds the tap-weight accumulator. This technique is used to force the gradient quantization error to be sufficiently noisy so the algorithm adaptation continues.

As a solution to the above problems, leakage terms are used in the sequential adaptation of the LMS algorithm. Leakage can be used to prevent the occurrence of coefficient drift, overflow in a limited precision environment, etc, by providing a compromise between minimizing the MSE and containing the energy in the adaptive filter impulse response. A cost function similar to Eq. (1.2) is minimized and is given by

$$J(n) = \min_{C(n)} < e^2(n) + \mu C(n)^T C(n) > \tag{1.3}$$

where μ is a positive control parameter (leakage).

Unfortunately, in some real time applications, the Leaky LMS(LLMS) adaptation scheme can be performed only after a delay. This delay comes from the fact that the desired signal and thus the error signal are available only after several symbol intervals. Such condition can be encountered in numerous cases e.g. in systolic array implementation [24] where the filter output is generated after a delay that equals the number of filter taps. It can also be found in decision directed adaptive equalization where the Viterbi algorithm is used [25]. This delay can have detrimental effects upon the regular LMS algorithm stability as shown in [3]. In this thesis, we study the effect of the delay upon the performance of the Leaky LMS algorithm in stationary environments. We will refer to this algorithm as the leaky delayed LMS (LDLMS) algorithm. The structure and feedback model for tap adjustments for LDLMS are shown in Figure 1.3.

In this simplified adaptive structure, the delay parameter is shown explicitly and it accounts for different situations where a delay can arise. So, the adaptation steps will be performed using inputs delayed by D symbols. Stability bounds, convergence behavior, and excess MSE for this new algorithm are to be investigated. Theoretical results are first derived and later verified by simulations. It will be shown that leakage combined with delay can provide compromise performance. Finally, examples of applications of the LDLMS algorithm are provided.

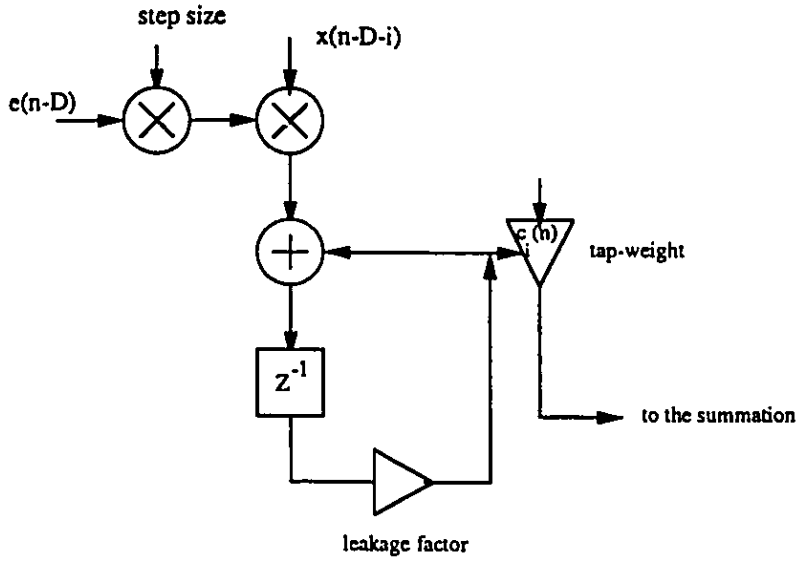


Figure 1.3: Structure of LLMS algorithm with delayed adjustments. (D denotes the delay parameter).

1.2 Thesis Presentation

This thesis is presented in five chapters. Chapter 1 includes the introduction of the problem and the suggested solutions. Chapter 2 reviews the LMS, LLMS, and the delayed LMS(DLMS) algorithm as well as other DLMS variants such as the normalized LMS algorithm with feedback delay. Mathematical analysis of these algorithms are provided. The LDLMS algorithm is introduced in Chapter 3. The effects of the delay, leakage, and the step size on the stability bounds of the LDLMS adaptive algorithm will be examined using the root loci techniques. Steady state excess MSE of the algorithm is derived and hence tighter bounds on the step size are found. These stability bounds are in terms of delay, leakage, and eigenvalue distribution. The stability bound is general in the sense

that other stability bounds of the LMS, LLMS, and DLMS algorithms can be easily derived. The derived bounds account for different types of input probability distribution functions. This is due to adaptive algorithms which react differently under different input distribution functions. In Chapter 4, we discuss LDLMS applications where a system identification problem is considered. In such applications, the LDLMS algorithm will prove to be a good means to remedy different situation problems such as drift problem. Finally, in Chapter 5, we include some conclusions and discussions about this work. Some suggestions for further research in the analysis and applications of the LDLMS algorithm will be provided.

Chapter 2

Review of the Least Mean-Squared Algorithms

The interest of adaptive filters continues to grow and many algorithm variants find real time applications in many areas. The least mean-squared adaptive algorithm has been widely used in adaptive filtering applications [13] because it requires less hardware and is therefore less costly. For instance, the Leaky LMS algorithm is used in fractionally spaced equalizers [8] and the delayed LMS algorithm is also used in multiprocessor implementations [34], [35].

In the following, we will review the basics of the LMS algorithm as well as the leaky LMS algorithm as a way of improving performance in non-ideal input cases. This review, as applied to a system identification problem, covers stability bound, convergence speed, and steady state excess MSE of the adaptive algorithm. The effect of the delay on the LMS algorithm will also be investigated. Some applications of these algorithms will be given as examples. Stability properties of the normalized LMS(NLMS) algorithm with feedback delay is also summarized. An exact analysis of delayed NLMS required no approximations and only an assumption of mixing conditions on the input signal is applied.

2.1 The Least Mean-Squared Algorithms

The least mean-squared(LMS) algorithm is a stochastic gradient version of the well known steepest descent (SD) optimization technique, driven with a fixed step size.

Let $\hat{\nabla}(n)$ designate the value of the gradient vector at time n and let $C(n)$ designate the value of the tap-weight vector at time n. According to the SD approach, the value of the tap-weight vector at time $n+1$ is updated by using the simple recursive relation:

$$C(n+1) = C(n) + \frac{1}{2}\beta[-\hat{\nabla}(n)] \tag{2.1}$$

where the tap-weight vector is defined as

$$C^T(n) = [c_1(n), c_2(n), \ldots, c_N(n)] \tag{2.2}$$

where β is the step size parameter. We consider the system model shown in Figure 1.2. Assuming that the tap-input vector $X(n)$ and the desired $d(n)$ are jointly stationary random processes, then the MSE $J(n)$, given by

$$\begin{aligned} J(n) &= < [d(n) - y(n)]^2 > \\ &= < e(n)^2 > \end{aligned} \tag{2.3}$$

is a quadratic function of the tap-weights. Equation (2.3) now can be written as

$$J(n) = \sigma_d^2 - 2C^T(n)P + C^T(n)HC(n) \tag{2.4}$$

where:

$\sigma_d^2 =< d^2(n) >$: the variance of the desired response $d(n)$

$P =< d(n)X(n) >$: cross-correlation vector between $d(n)$ and $X(n)$

$H =< X^T(n)X(n) >$: correlation matrix of the tap-input $X(n)$.

To get the optimum value C_{opt} for the tap-weight vector, we differentiate (2.4) with respect to the weight vector $C(n)$ and equate it to zero. Thus, we obtain

$$\begin{aligned} \nabla(n) &= \frac{\partial J(n)}{\partial C(n)} \\ &= -2P + 2C_{opt}^T H = 0 \\ &\Rightarrow C_{opt} = H^{-1}P. \end{aligned} \tag{2.5}$$

If we substitute the above equation into equation (2.4), then the minimum MSE is found to be

$$J_{min} = \sigma_d^2 - PH^{-1}P. \tag{2.6}$$

Practically, it is not possible to make exact measurements of the gradient vector at each instant, therefore the gradient vector must be estimated from the available data. The same holds for H. The LMS algorithm thus uses the estimated H and P. The simplest way of estimating H and P, is to use instantaneous estimates that are based on sample values of the tap input vector and the desired response. These estimates are given as follows:

$$\begin{aligned} \hat{H}(n) &= X(n)X^T(n) \\ \hat{P}(n) &= X(n)d(n). \end{aligned} \tag{2.7}$$

where the input vector is defined as

$$X(n)^T = [x(n), x(n-1), \ldots, x(n-N+1)] \tag{2.8}$$

So, the estimated gradient is

$$\begin{aligned} \hat{\nabla}(n) &= -2X(n)d(n) + 2X(n)X^T(n)C(n) \\ &= -2e(n)X(n) \end{aligned} \tag{2.9}$$

Substituting equation (2.1) into equation (2.9), the updated tap-weight vector becomes

$$C(n+1) = C(n) + \beta X(n)e(n), \tag{2.10}$$

where

$$e(n) = d(n) - X^T(n)C(n) \tag{2.11}$$

The procedure described for the derivation of the LMS algorithm may be viewed as an approximation of the SD algorithm using instantaneous estimates of the correlation matrix $\hat{H}(n)$ and the cross-correlation vector $\hat{P}(n)$. The LMS gradient is an unbiased estimate of the the SD gradient. In other words,

$$\lim_{n \to \infty} < C(n) >= C_{opt}. \tag{2.12}$$

Throughout this work, several fundamental assumptions are used to make the analysis of the LMS algorithm mathematically tractable. These assumptions are:

1. The input process is statistically independent of all previous samples

$$< X(n)X^T(k) > \ = 0, \quad k = 0, 1, \ldots, n-1.$$

2. The input process at sample time n is statistically independent of all previous samples of the desired signal $d(k)$,

$$< X(n)d(k) > \ = 0, \quad k = 0, 1, \ldots, n-1$$

In view of point 1 and 2 of the above assumptions, the tap-weight vector $C(n+1)$ is independent of $X(n+1)$ and $d(n+1)$. However, from the LMS update equation, $C(n+1)$ depends on the previous samples of the input and the desired signals. Also, $C(n+1)$ depends on the initial starting value $C(0)$ of the adjustment process.

2.1.1 Average Tap-Weight Vector of the LMS Algorithm

Let's define the tap-weight error vector of the LMS algorithm as follows:

$$\epsilon(n) = C(n) - C_{opt}, \tag{2.13}$$

where C_{opt} is the optimum Wiener solution.
Using (2.10) and (2.13) and the fundamental assumptions stated earlier, the time progression of the tap-weight error can be rewritten as [13]

$$< \epsilon(n+1) >= (I - \beta H) < \epsilon(n) >, \tag{2.14}$$

where $< \epsilon(n) >$ is the average tap-weight error vector. We use the fact that H is symmetric and it can be represented in the form of $H = V\Lambda V$. Λ is the diagonal eigenvalue matrix of H; and since H is positive definite; $V$is the orthonormal matrix whose i^{th} column is the eigenvector V_i of H associated with λ_i. So, applying this coordinate transformation to

(2.14), we deduce that a necessary and a sufficient condition of convergence of $< \epsilon(n) >$ to zero when $n \rightarrow \infty$ is

$$0 < \beta < \frac{2}{\lambda_{max}}, \tag{2.15}$$

where λ_{max} is the largest eigenvalue of the correlation matrix H. Under this condition, the LMS algorithm is said to be convergent in the mean.

2.1.2 Average Mean-Squared Error of the LMS Algorithm

Another approach to derive a bound on the step size to guarantee stability of the LMS algorithm based on the average MSE $< e^2(n) >$ is utilized. The MSE in equation (2.4) is written in terms of the tap-weight error vector $\epsilon(n)$ as follows:

$$J(n) = J_{min} + \epsilon(n)^T H \epsilon(n) \tag{2.16}$$

The first term of the above equation is the minimum MSE when $C(n)$ approaches its optimal value C_{opt}. The second term results from the noisy estimates of the gradient and it is known as the excess MSE. The excess MSE is defined as

$$\begin{aligned} J_{ex}(n) &= J(n) - J_{min} \\ &= \epsilon(n)^T H \epsilon(n). \end{aligned} \tag{2.17}$$

If the elements of the vector $X(n)$ are independent and have identical even distributions with variance σ_x^2, the kurtosis ν_x for zero mean variable x is defined as

$$\nu_x = \frac{< x_n^4 >}{< x_n^2 >^2}. \tag{2.18}$$

then Eq. (2.17) can be rewritten as [27]:

$$J_{ex}(n+1) = mJ_{ex}(n) + h, \tag{2.19}$$

where m and h are defined as:

$$\begin{cases} m = 1 - 2\beta\sigma_x^2 + \beta^2\sigma_x^4(N + \nu_x - 1) \\ h = J_{min}(\beta\sigma_x^2)^2 N. \end{cases}$$

The kurtosis, ν_x which is defined above by $\nu_x = \frac{<x^4>}{<x^2>^2}$ is different for each distribution process. Its maximal value, $\nu_x = \infty$, is for obtained for the Cauchy distribution and its minimal value, $\nu_x = 1$, is obtained for binary symmetric distribution [27]. However, for the Gaussian distribution function, $\nu_x = 3$. The kurtosis ν_x is included in the above equation since for each density distribution function, the stability bounds of the adaptive algorithms are different. If the filter converges, we get the steady state excess MSE as

$$J_{ex}(\infty) = \frac{J_{min}\beta\sigma_x^2 N}{2 - \beta\sigma_x^2(N + \nu_x - 1)}. \tag{2.20}$$

The values of the step size β for which the denominator of (2.20) is zero denotes the stability bounds of the adaptive algorithm on the mean-squared sense. Thus, the upper bound on the step size is

$$0 < \beta < \frac{2}{\sigma_x^2(N + \nu_x - 1)}. \tag{2.21}$$

The derivation of the two above expressions can be found in [27].

The step size β, the number of taps N, and the eigenvalues of the input autocorrelation matrix H are the major parameters that affect the response of the LMS adaptive algorithm. For a small value of β, slow convergence of both $C(n)$ towards its optimum and the MSE $J(n)$ towards its J_{min} result. Moreover, a small amount of $J_{ex}(\infty)$ results. The opposite can be concluded when the step size β is large.

The eigenvalue spread of H affects the convergence speed of $< e^2(n) >$. For eigenvalue spread, the LMS algorithm slows down its convergence and a large number of iterations is required [13]. In addition, the convergence of $< \epsilon(n) >$ is much more affected by the eigenvalue spread than $< e^2(n) >$.

The convergence properties of (2.21) depend on the number of taps, which is not the case in (2.15). So, the convergence conditions of $< e^2(n) >$ is much tighter than the one imposed by equation (2.15). It can be seen from (2.15) and (2.21) that

$$\lambda_{max} < \sigma_x^2(N + \nu_x - 1). \tag{2.22}$$

Therefore, there are values of β for which $< \epsilon(n) >$ converges but $< e^2(n) >$ does not. By choosing β from equation (2.21), convergence condition of $\epsilon(n) >$ will be automatically

satisfied. When both quantities converge, $< \epsilon(n) >$ converges much faster than $< e^2(n) >$. The stability bound in equation (2.21) is more practical than in equation (2.15) since for most applications cases N will be the dominating factor.

2.2 Leakage in Adaptive Algorithms

Introducing leakage in adaptive algorithms has been proven to be useful in numerous applications such as channel equalizers and ADPCM codecs for telephone transmission. For example, it is introduced in ADPCM systems to improve the robustness of the adaptation to transmission errors. It has also been shown to be robust in counteracting many problems such as overflow in finite-precision implementation, weak excitation of input signals and other stability problems. It is known that leakage is used in the least mean squared algorithms (LMS) and its variants and in the least squares algorithms such as fast Kalman and FTF algorithms. The latter case will not be reviewed in this thesis and a discussion on it can be found in [22]. In this section, we first give a review of the LLMS algorithm theory and then illustrate some of its applications.

2.2.1 The Leaky LMS Algorithm

Some or all the tap weights can reach unacceptable values when the LMS algorithm is used under non-ideal situations. Either of the following cost functions has been minimized as a way to control the large tap build-up [8]:

$$J_1 = J + \mu \parallel C \parallel^2 \tag{2.23}$$

$$J_2 = J + \mu \parallel C \parallel \tag{2.24}$$

where J is the MSE, μ is positive real number much smaller than one, and $\parallel . \parallel$ denotes the norm of a vector. As it can be seen from equations (2.23) and (2.24), J_1 and J_2 attribute quadratic and magnitude penalties, respectively to the magnitude of the tap. In a manner similar to the commonly used estimated gradient algorithms, the update

equation of the LLMS is constructed so as to minimize the incremented instantaneous square error, $< e(n)^2+\mu C(n)^T C(n) >$. Then, the resultant update equation is given by

$$C(n+1) = C(n) + \beta e(n)X(n) - \beta\mu C(n), \tag{2.25}$$

where $C(n)$ is the tap-weight vector, $X(n)$ is the input vector, $e(n)$ is the error output, β and μ are the step size and the leakage parameters, respectively. For $\mu = 0$, (2.25) corresponds to the regular LMS. The operation of this algorithm is evident, the coefficient vector is multiplied by a constant slightly less than unity at each iteration before adding the correction term. When the coefficient vector is wandering in the direction of an eigenvector corresponding to a small eigenvalue, the leakage limits the size of the vector over time. The LLMS may be also written in different forms as follows:

$$C(n+1) = \gamma C(n) + \beta e(n)X(n) \tag{2.26}$$

or

$$C(n+1) = C(n) + \beta e(n)X(n) - \phi C(n), \tag{2.27}$$

where ϕ and γ are given by:

$$\begin{cases} \phi = \beta\mu \\ \gamma = 1 - \phi \end{cases}$$

Equations (2.26) and (2.27) are in fact the same, as only the leakage parameter in the updated equation is differently stated. An adaptive scheme of the LLMS algorithm using (2.27) was shown in Figure 1.3 with $D = 0$.

The effect of the leakage can be evaluated directly by using the orthonormal matrix V, thus the rotated tap-weight vector becomes $C'(n) = VC(n)$. The update equation for the rotated coefficients vector is given by:

$$\begin{aligned} C'(n+1) &= (1 - \mu\beta)C'(n) - \beta(-P' + \Lambda C'(n)) \\ &= (1 - \mu\beta I - \beta\Lambda)C'(n) + \beta P' \\ &= (I - \beta(\mu I + \Lambda))C'(n) + \beta P', \end{aligned} \tag{2.28}$$

where $P' = VP$ is the rotated cross-correlation vector and Λ is the eigenvalue matrix of H. In equation (2.28), the following term can be modified to

$$I - \beta(\mu I + \Lambda) = I - \beta\Lambda_{new}, \tag{2.29}$$

where $\Lambda_{new} = \mu I + \Lambda$. In this case, equation (2.28) can be rewritten as

$$C'(n+1) = (I - \beta\Lambda_{new})C'(n) + \beta P'. \tag{2.30}$$

If we use the coordinate transformation introduced before, we obtain

$$H_{new} = \mu I + H. \tag{2.31}$$

From above, we see that the effect of leakage is to modify the eigenvalues and the correlation input matrix.

If we subtract the true optimum weight vector C_{opt} from equation (2.25) and solve for the steady state tap-weight error vector, we find

$$< \epsilon(\infty) > \; = \; \sum_{i=0}^{N-1} \frac{\mu}{(\lambda_i + \mu)} \frac{V_i^T P}{\lambda_i} V_i \tag{2.32}$$

The effect of the presence of the leakage in Λ_{new} is to eliminate the catastrophic effects that can accompany small or zero eigenvalues. Equation (2.32) represents the penalty which comes from minimizing the modified gradient algorithm instead of the true gradient J. The magnitude of $< \epsilon(\infty) >$ in (2.32) can be made reasonably small by the proper choice of μ. An expression of the LLMS steady state excess MSE for a binary input signal was derived in [8]. The expression is given here by

$$J_{ex}(\infty) = \frac{\beta\lambda_{max} N\sigma_x^2 J_{min} + \mu^2 P H^{-1} P}{2\lambda_{avg} + \mu(1 - \beta\lambda_{avg}) - \beta(\lambda_{max} N\sigma_x^2 + \mu^2)} \tag{2.33}$$

where:

λ_{avg} is the average value of the autocorrelation matrix H,

λ_M is the maximum eigenvalue of H, and

σ_x^2 is the variance of the input data.

The above equation shows that for a small value of μ, the effect of leakage on the minimum MSE becomes minimal. The bias on the true optimum tap-weight vector can be controlled by the leakage value. To summarize, the introduction of the leakage leads to the following desirable situations.

- All eigenvalues of the updated equation are positive even if some of the input eigenvalues are zero.
- The positivity of all eigenvalues implies a unique solution and bounded time constants.
- The overall algorithm will always converge with finite time constants. Its time constant is bounded by

$$\begin{aligned} \tau_{max} &\approx \frac{1}{\beta\lambda_{new}} \\ &= \frac{1}{\beta\mu + \beta\lambda} \\ &\leq \frac{1}{\mu\beta} \end{aligned} \tag{2.34}$$

while for the case of the LMS algorithm, the time constant is bounded by

$$\tau_{max} \approx \frac{1}{\beta\lambda} \tag{2.35}$$

2.3 Digital Implementation of the LLMS Algorithm

In finite-precision implementation of adaptive algorithms, a bias on the true optimum tap-weight vector C_{opt} results. The effects of the resulting bias on the performance of the LLMS algorithm can be well visualized from the steady state tap-weight error and the steady state excess MSE.

Let's denote this bias by the quantization noise vector **q**. The vector **q** has a magnitude q which is less than half of the uniform quantization interval. It will be shown that the implementation of the LMS algorithm without leakage can produce degradation

effects such as overflow in registers when some of the eigenvalues of the input autocorrelation matrix are very small or negligible. When implementing the LLMS, overflow in registers or drifting phenomena can be prevented.

The adjustment tap-weight algorithm with the digital bias q is given by:

$$C(n+1) = C(n) - \beta\,[e(n)X_n + \mathbf{q})]\,. \tag{2.36}$$

The mean tap-weight error vector is given in [8] by

$$< \epsilon(n) >= H^{-1}\mathbf{q} \;\; when\; n \rightarrow \infty \tag{2.37}$$

If we consider the i^{th} eigenvalue λ_i and i^{th} eigenvector V_i of H we get

$$< \epsilon(n) >= \sum_{i=0}^{N-1} \frac{V_i^T \mathbf{q} V_i}{\lambda_i} \;\; when\; n \rightarrow \infty \tag{2.38}$$

If a small eigenvalue λ_i has eigenvector V_i which is not orthogonal to q, the steady state tap error in (2.38) can be quite significant.

The effect of q on the steady state MSE can be examined by considering the size of the residual MSE

$$J_{ex}(n) =< \epsilon(n)^T H \epsilon(n) > . \tag{2.39}$$

Following a similar procedure in [7] and using the fundamental standard assumptions, the steady state excess MSE can be described as

$$J_{ex}(\infty) \;\; = \;\; \frac{\beta\lambda_{max}(N)\{\sigma_x^2 J_{min} + q^2\}}{2\lambda_{avg} - \beta\lambda_{max}(N)\sigma_x^2}, \tag{2.40}$$

where σ_x^2 is the input power and λ_{max} is the largest eigenvalue of H. As it can be seen, the effect of the bias q on $J_{ex}(\infty)$ is negligible. However, a build-up in one or more of the tap-weights can result.

On the other hand, when using the LLMS algorithm,

$$C(n+1) = C(n) + \beta(e(n)X_n + \mathbf{q}) - \beta\mu C(n), \tag{2.41}$$

adjustments are made to minimize the following minimum MSE,

$$J = C^T(n)RC(n) - 2C^T(n)P + 1, \tag{2.42}$$

where $R = H + \mu I$. Obviously, both R and H have the same eigenvectors. However, the eigenvalues of R are $(\lambda_i + \mu)$. In this case, the optimum tap-weight vector is a function of μ and is given by

$$C_\mu = (H + \mu I)^{-1} P \tag{2.43}$$

In a manner similar to the computation of (2.38), the steady state tap-weight error vector can be computed and is given by

$$< \epsilon(\infty) >= \sum_{i=1}^{N} \frac{V_i^T \mathbf{q} V_i}{\lambda_i + \mu} + \mu \sum_{i=1}^{N} \frac{V_i^T P V_i}{\lambda_i(\lambda_i + \mu)} \tag{2.44}$$

We can see the first right hand term of the above equation is similar to equation (2.38) except the eigenvalues have now been modified to guarantee no negligible or nonzero eigenvalues. The second term is proportional to μ and it results from the minimization of the modified MSE.

The steady state excess MSE in (2.41) is now derived with the presence of leakage in quite a similar manner and is given by

$$J_{ex}(\infty) = \frac{\beta \lambda_{max}(N) \sigma_x^2 J_{min} + \mathbf{q}^T \mathbf{q} + 2\beta\mu \mathbf{q}^T P + \mu^2 P^T H^{-1} P}{2\lambda_{avg} - \beta \lambda_{max}(N) \sigma_x^2} \tag{2.45}$$

We note that the leakage parameter in equation (2.45) does not introduce considerable degradation if it is chosen in the order of a smallest eigenvalue λ_{min}. In this case $J_{ex}(\infty)$ can be almost unaffected by μ. Also, by proper control of the leakage value in (2.44), the second term can be made to be very small. Thus a compromise has to be obtained between these two quantities in (2.44) and (2.45).

2.4 Application of the Leaky LMS Algorithm to Digitally Implemented Fractionally Spaced Equalizers.

One of the applications of the LLMS algorithm is in the Fractional Spaced Equalizer (FSE). These equalizers are nonrecursive adaptive transversal filters whose tap weights are spaced a fraction of a symbol time apart. They provide significant performance for modem under severe linear distortions [33]. For instance, FSE is very effective over the conventional synchronous equalizer in compensating for a channel with poor envelope delay characteristics. However, when FSE is digitally implemented, it was observed that after a period of time, some of the taps would become very large. Consequently, some of the registers which compute the partial sums of the equalizer output will overflow and hence lead to stability problems. This build-up in the tap-weight results from the ill-conditioning of the channel correlation matrix which influences the rate of convergence of equalizer taps to those optimum values. Moreover, the contours of equal new squared error in the FSE are ellipses while their eccentricity depends on the eigenvalue distribution. In [33], it is shown that for infinitely long FSE equalizers, half the eigenvalues are zero. The operation of the FSE equalizers can be stabilized by using the LLMS adaptive algorithm instead of the conventional estimated gradient algorithm (LMS).

Section 2.3 is useful in explaining the FSE performance in term of its final taap-weight optimum settings and its equalized MSE. In (2.37), if we assume one tap equalizer, $< \epsilon(n) >$ will be directly proportional to the bias and definitely no build up occurs.

The equalized MSE can be examined by looking at equation (2.40). In the case of the LMS algorithm, it can be easily seen that the effect of $\mathbf{q}$ on J_{ex} is negligible though there can be, on the average, a build-up of one or more tap-weights.

On the other hand, When the FSE is operated by the LLMS, the average tap-weight error vector is given by (2.44) while the final excess MSE is now identical to equation (2.45). From these two expressions, we can see how the eigenvalues have been modified

to avoid any catastrophic results that can occur from vanishing small eigenvalues. Leakage has to be chosen sufficiently large to properly control the magnitude of the first term of equation (2.44) and sufficiently small to not increase $< \epsilon(\infty) >$ through the second term in (2.44).

Finally, effective control of tap-drifting for FSE was tested in laboratory experiments by using the LLMS algorithm and its performance was reported successfully in [8]. The range of μ that gave better performance was between 2^{-7} and 2^{-10} with the step size β of 2^{-11}. It was observed that leakage values less 2^{-6} introduce negligible degradation in the FSE performance.

2.5 Delayed Least Mean-Squared Algorithm

In many applications of adaptive filtering, a delay in the coefficient update is imposed on the LMS algorithm. This is due to the error feedback which may not be available till several symbols later. For example, in data echo cancellation the far-end data signal slows down the adaptation convergence. To speed up the adaptation convergence, the data signal is removed from the cancellation error in decision-directive fashion [32]. This is illustrated in Figure 2.1. Since the receiver decision circuit includes delay, a compensating delay is incorporated in the other inputs to the echo cancellor. Thus the adaptation scheme operates on a delayed basis. Another example is in situations where high sampling rates or large number of filter taps are needed. The computational demand imposed on a single processor becomes prohibitive, implementations using multiple processors are required. However, a critical limitation on the throughput of possible implementation is imposed by the feedback error of the LMS. By inserting delay in the LMS algorithm, the resulting delayed LMS(DLMS) algorithm has been shown to guarantee stable convergence properties assuming an appropriate step size is selected [3]. In the following sections, we review the theory of DLMS algorithm and its performance in terms of convergence and stability bounds as given in [4]. Other approaches and DLMS variants will be also included.

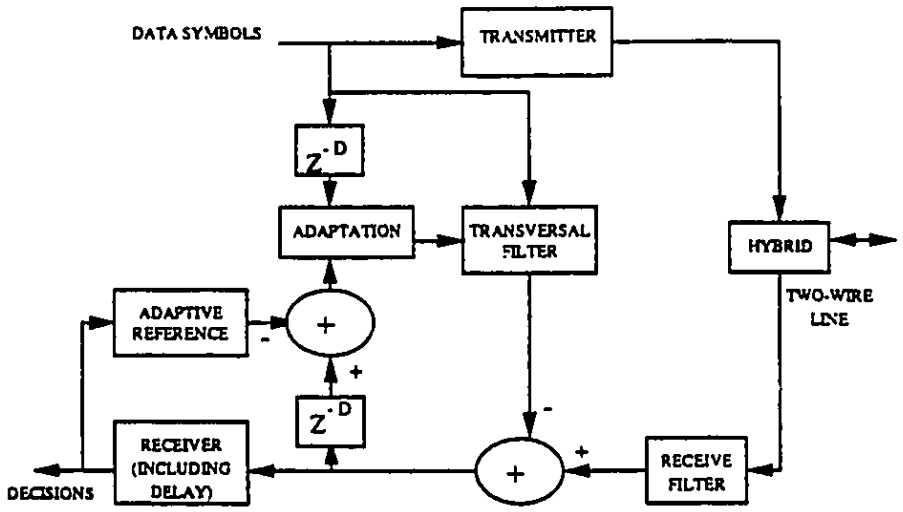


Figure 2.1: Baseband adaptive reference echo canceller with delay

2.5.1 System Equation of DLMS Algorithm

The LMS with a delay is described by the following equations:

$$C(n) = C(n-1) + \beta e(n-D)X(n-D) \tag{2.46}$$

where the feedback error and the estimated signal are given by

$$e(n-D) = d(n-D) - y(n-D) \tag{2.47}$$

$$y(n-D) = X^T(n-D)C(n-D-1). \tag{2.48}$$

The delay incorporated in (2.46) does not alter the criterion for optimality which results from the minimization of the MSE $< e^2(n) >$. The $< . >$ denotes ensemble averaging over the statistics of the signal as previously defined.

The optimum coefficient vector C_{opt} which minimizes the MSE is exactly the same as the one in LMS algorithm and is given by

$$C_{opt} = H^{-1}P, \tag{2.49}$$

where H and P have been defined previously in section 2.1.

As before it is assumed that $X(n)$, $\epsilon(n)$, and $e_{opt}(n)$ are statistically independent. The MSE can be expressed as

$$J(n) = J_{min} + J_{ex}(n-1) \tag{2.50}$$

where $J_{min} = \sigma_d^2 - P^T H^{-1} P$ and the excess MSE at time n is defined as

$$J_{ex}(n-1) = < \epsilon^T(n-1) H \epsilon^T(n-1) >, \tag{2.51}$$

where $\epsilon(n) = C(n) - C_{opt}$.

2.5.2 Error Convergence and Stability Bounds of DLMS Algorithm

The steady state excess MSE was obtained as in [4]

$$J_{ex}(\infty) \cong \frac{J_{min}\alpha N \beta \sigma_x^2}{2 - (\alpha M + 2D)\sigma_x^2\beta + D(D+1)\sigma_x^4\beta^2 - \frac{1}{3}D(D+1)(2D+1)\sigma_x^6\beta^3 + \ldots} \tag{2.52}$$

where:

N is the filter length, and α is the ratio of root mean square λ_{rms} of the eigenvalues of the autocorrelation matrix H to the average eigenvalue λ_{avg}. The delay is a positive integer value and denoted by D and $M = N + \nu_x - 1$, where ν_x is previously defined in (2.18). The denominator of (2.52) has an infinite number of terms. However, in the region of stability, the step size is normally small. Hence, higher order terms decay fast. The stability bound on the step size is found by equating the denominator of (2.52) to zero and is given by

$$g(b) = 2 - (\alpha M + 2D)b + D(D+1)b^2 - D(D+1)(\frac{1}{3}(2D+1)b^3 + \ldots = 0, \tag{2.53}$$

where $b = \beta\sigma_x^2$. As b increases from zero to a value b_{max}, $g(b)$ decreases monotonically from $g(0) = 2$ to $g(b_{max})$ where $J_{ex}(\infty)$ at b_{max} blows up.

For applications where $\beta D << 1$, $g(b)$ can be approximated as

$$g_1(b) = 2 - (\alpha M + 2D)b. \tag{2.54}$$

The steady state excess MSE is now given by

$$J_{ex}(\infty) \cong \frac{J_{min}\alpha N b}{2 - (\alpha M + 2D)b}. \tag{2.55}$$

In this case, the upper bound on the step size can be easily found as

$$b_{max} = \frac{2}{\alpha M + 2D}$$

or

$$\beta_{max} = \frac{2}{\sigma_x^2(\alpha M + 2D)} \tag{2.56}$$

However, for relatively large values D, the second order term has to be retained. Thus, $g(b)$ is reduced to

$$g_2(b) = 2 - (\alpha M + 2D)b + D(D+1)b^2. \tag{2.57}$$

The steady state excess MSE becomes

$$J_{ex}(\infty) \cong \frac{J_{min}\alpha N b}{2 - (\alpha M + 2D)b + D(D+1)b^2} \tag{2.58}$$

So, the maximum step size which constitutes the upper bound is described by

$$b_{max} = \frac{\alpha M + 2D - \sqrt{\alpha M + 2D^2 - 8D(D+1)}}{2D(D+1)}$$

or

$$\beta_{max} = \frac{\alpha M + 2D - \sqrt{\alpha M + 2D^2 - 8D(D+1)}}{\sigma_x^2 2D(D+1)} \tag{2.59}$$

The 3^{rd} order term may have to be included depending on how βD approaches unity. Then, numerical methods have to be employed to solve the following equation,

$$g_3(b) \approx 2 - (\alpha M + 2D)b + D(D+1)b^2 - \frac{1}{3}D(D+1)(2D+1)b^3. \tag{2.60}$$

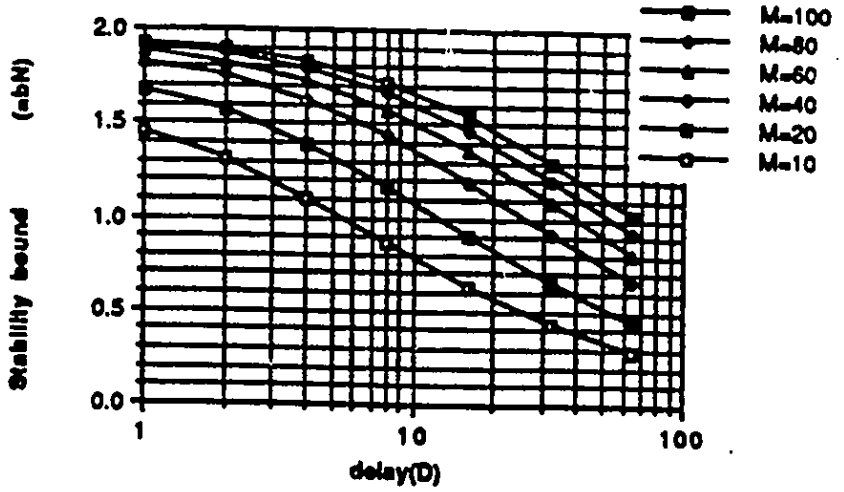


Figure 2.2: Stability bounds of the DLMS algorithm for different values of M ($\alpha = 1$); Figure from [4].

The stability bounds for different filter lengths are depicted in Figure 2.2 (here α and μ assumed 1 and 3, respectively).

It is shown in [4] that when $\frac{D}{\alpha M}$ is small, all approximations $g_1(b)$, $g_2(b)$, and $g_3(b)$ are close. However, when D gets bigger, the $g_1(b)$ approximation becomes unacceptable. Nevertheless, in practice, $Db \ll 1$ is usually chosen to avoid large excess mean-squared error.

The convergence behavior of DLMS depends on the roots of its characteristic equation given in [4]. Unlike the regular LMS algorithm, the roots of the characteristic equation of the DLMS system are not necessarily real. So, the decrease of the MSE is not necessarily monotonic. When D increases, the complex root becomes more apparent and the oscillatory behavior of the DLMS starts to appear especially during initial periods. Eventually the performance of the DLMS algorithm will be sufficiently degraded. Small delay introduces only a slight degradation in the convergence speed and the steady state misadjustment when an appropriate step size is used. However, the delay in updating the filter coefficients can have major effects in nonstationary environment tracking

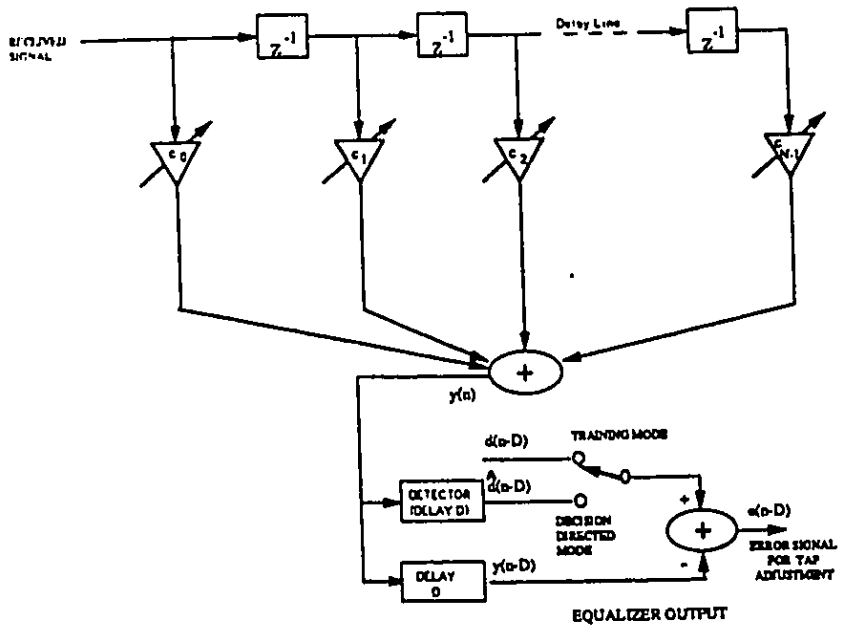


Figure 2.3: Adaptive equalizer with delayed decisions

capabilities.

2.6 Coefficient Convergence of the DLMS Algorithm

Another approach to derive the stability bound of the DLMS algorithm based on its coefficient convergence was considered by Peter Kabal in [6]. The derived expression is a necessary and sufficient condition for the stability and convergence of the coefficients of the DLMS algorithm. The model of DLMS algorithm considered was an adaptive equalizer with decision directive mode as shown in Figure 2.3. The tap-weight adjustment algorithm is identical to (2.46). The rotated tap-weight error vector $W(n)$ of the DLMS can be

given using the coordinate transformation techniques on the tap-weight error vector $\epsilon(n)$. By taking the Z-transform of the i^{th} tap-coefficient error $w_i(n)$, we obtain

$$W_i(z) = \frac{z^{D+1} w_i^{(o)}}{z^{D+1} - z^D + \beta\lambda_i} \tag{2.61}$$

where $w_i^{(o)}$ is the rotated tap-weight error at instant $n = 0$. The poles of $W_i(z)$ have to be within unit circle so that the stability criteria can be fulfilled. In other words, the characteristic polynomial which is defined by

$$F_i(z) = z^{D+1} - z^D + \beta\lambda_i \tag{2.62}$$

has to have all the roots for each i lie within unit circle. Determining the location of the roots of $F_i(z)$ and solving for β, the range values of the step size for which the adjustment algorithm is stable and converges is

$$0 < \beta < \frac{2}{\lambda_j} sin\left[\frac{\pi}{2(2D+1)}\right]. \tag{2.63}$$

The stability bound has to be satisfied for each i. So, for all modes of the adaptive filter, the necessary and sufficient condition of convergence of all the tap-weights on the mean is given by

$$0 < \beta < \frac{2}{\lambda_{max}} sin\left[\frac{\pi}{2(2D+1)}\right], \tag{2.64}$$

where λ_{max} is the largest eigenvalue of the correlation matrix H. The condition on the convergence of DLMS algorithm can be reduced to that of the LMS by putting $D = 0$.

As it can be seen, a tradeoff between the rate of convergence and stability exists in (2.64). For small values of the step size, the system has an exponential response with time constant given by,

$$\tau = ln\frac{1}{1-r}, \tag{2.65}$$

where r is the dominant real root within unit circle and which satisfies the following equation

$$\beta\lambda_{max} = r^D(1-r) \tag{2.66}$$

However, for larger D, more than one root begin to appear. If the magnitude of the complex roots dominates, then the system response becomes oscillatory and degradation of the DLMS performance starts to appear.

2.7 Normalized LMS algorithm with a Feedback Delay

The convergence properties of the standard normalized LMS (NLMS) algorithm are well established and understood in [30, 31]. The NLMS algorithm ensures that the step size does not become too large thus stability is guaranteed. The time-dependent step size $\beta(n)$ is defined by $\beta(n) = \frac{\eta}{\|X(n)\|^2}$, where η is the normalized step size chosen between 0 and 2. The NLMS is then given by the

$$C(n+1) = C(n) + \eta \frac{e(n)X(n)}{\|X(n)\|^2}. \tag{2.67}$$

We should mention that the input normalization does not prevent the algorithm from drifting. Also, the inclusion of $\|X(n)\|^2$ increases the computational complexity.

In a situation where a delay is introduced in the update equation of the NLMS algorithm, the norm of the weight error vector can increase or decrease at any given time update. For this reason, stability and convergence criteria of the NLMS algorithm have been examined anew. It was shown in [28] that for any delay, D, the gain parameter, β, can be selected to be sufficiently small so that exponential convergence of the tap-weight error vector $\epsilon(n)$ is guaranteed. The mixing condition on the input is assumed so that the exponential convergence of the delayed NLMS(NDLMS) algorithm is imposed. We will start by defining the mixing concept and then deriving the stability bounds of the NDLMS algorithm.

2.7.1 Mixing Condition

When the mixing condition to the inputs is applied for the NLMS algorithm, stationarity or periodicity property on the input signal are not necessary requirements for the

study of the convergence and stability bounds of the algorithm. The input can be either stochastic or deterministic. The mixing condition is said to be satisfied if [29] there exists some integer K and some $\varrho > 0$ such that

$$\lambda_{min}(\frac{1}{K}\sum_{i=n}^{n+K-1}\frac{X(i)X^T(i)}{\|X(i)\|^2}) \geq \varrho \tag{2.68}$$

for all integer values of n. The λ_{min} stands for the smallest eigenvalue of the autocorrelation matrix and is given by

$$\lambda_{min} \quad = \quad \textit{smallest eigenvalue of } \frac{1}{K}\sum_{i=n}^{n+K-1} X(i)X^T(i). \tag{2.69}$$

2.7.2 The NDLMS algorithm

The NDLMS algorithm becomes

$$C(n+1) = C(n) + \eta\frac{e(n-D)X(n-D)}{\|X(n-D)\|^2}. \tag{2.70}$$

The above algorithm is a variant of the DLMS in which a squared norm of the input samples scales down the DLMS correction term.

If we introduce the tap-weight error vector, $\epsilon(n)$, in (2.70) gives as

$$\begin{aligned}\epsilon(n+1) \quad &= \quad \epsilon(n) - \eta\frac{X(n-D)X^T(n-D)}{\|X(n-D)\|^2}\epsilon(n-D) \\ &\quad +\eta\frac{d(n-D) - X^T(n-D)C_{opt}}{\|X(n-D)\|^2}\end{aligned} \tag{2.71}$$

The stability condition of NDLMS algorithm is investigated via the homogeneous part of (2.71), i.e.

$$\epsilon(n+1) = \epsilon(n) - \eta\frac{X(n-D)X^T(n-D)}{\|X(n-D)\|^2}.\epsilon(n-D) \tag{2.72}$$

The proof of the exponential convergence for the NDLMS follows a similar procedure used for the NLMS under mixing input by Wess and Mitra [29]. They showed that $\epsilon(n)$ of NLMS algorithm satisfies the following:

$$\|\epsilon(n+K)\|^2 \quad \leq \quad [1 - \frac{2\varrho\eta K(2-\eta)}{2+2\eta(K-1)+\eta^2K(K-1)}]\|\epsilon(n)\|^2 \tag{2.73}$$

The factor multiplying $\|\epsilon(n)\|^2$ in (2.73) has to be less than one so that an appropriate value for η can be found. This value will guarantee the convergence of the NLMS algorithm. In [29], a term is added to the (2.73) that accounts for the presence of the delay in the updated equation of NLMS algorithm. Thus, (2.73) can be rewritten as

$$\|\epsilon(n+K)\| \leq \delta Y \tag{2.74}$$

where

$$\delta = \sqrt{1 - \frac{2\varrho\eta K(2-\eta)}{2+2\eta(K-1)+\eta^2 K(K-1)}} + \eta^2 KD \tag{2.75}$$

and Y denotes an upper bound on the finite set of vectors $\epsilon(n-2D)$, $\epsilon(n-2D+1)$,...,$\epsilon(n+K)$.

From that, η should be chosen small enough to insure that $\delta < 1$. The final result is obtained for :

$$\epsilon(i) \leq \delta Y \quad , \quad n+K \leq i \leq n+2K+2D-1 \tag{2.76}$$

In other words, given any set of $K+2D$ consecutive vectors, which have some bound Y, the next $K+2D$ vectors will have bound δY. The next set vectors will be bounded by $\delta^2 Y$, and so on. Therefore, this will prove the exponential convergence of $\epsilon(n)$ to zero. This analysis did not require any approximation but it only used the assumption of mixing condition on the input process.

2.8 Conclusion

In this chapter, a theoretical review of the convergence and stability bounds of the LMS algorithm as well as of its variants, the LLMS algorithm, the DLMS algorithm, and the NDLMS algorithm, were provided. The leakage algorithm has been shown to be robust to the drift problem and it was successfully used in different applications as channel equalizer and echo cancellations. The value of leakage had to be much smaller than one not to affect optimum values. It was also shown that in certain applications a delay is incorporated in the updated LMS algorithm. This delay may result from various situations including

hardware time-processing, parallel implementation of LMS algorithms and in decision directed adaptive equalization. The stability bounds of DLMS were reviewed and shown to be much tighter than in the LMS case. The accuracy of the approximation orders of the stability bounds on the step size depends mainly on the ratio $\frac{D}{\alpha M}$ ratio. The NDLMS algorithm with the mixing condition on the input signal was reviewed and its stability bound was derived without any approximation. It was shown that the step size has to be chosen sufficiently small to guarantee the criteria imposed on the input process.

Chapter 3

The Leaky Delayed Least Mean-Squared Algorithm

The leakage LMS(LLMS)algorithm was shown to provide improved performance compared to the standard LMS for various applications [8]. In Chapter 2, use of leakage in the LMS to prevent filter coefficient drifting was discussed [10] along with application in fractionally spaced adaptive equalizers(FSE) for telephone data modems [5]. The inherent delay in updating the coefficients is still a problem for the LLMS. In this chapter, a study of the effects of delay on the performance of the LLMS algorithm is provided. The leaky delayed LMS(LDLMS) algorithm is studied in a system identification set up and the expression for the steady state excess MSE is derived for a stationary input process. Bounds on step size are also computed as a function of the leakage parameter μ. The formula derived for the excess steady state MSE is in fact a general expression from which the bounds on step size of LMS, DLMS, and LLMS can be obtained. The analytical results for derived step size bounds are checked by computer simulations. Effects of step size, leakage, filter length, and delay on the steady state excess MSE and on convergence speed are also investigated in detail. It was shown in [3] that as the delay in DLMS increases, the convergence speed slows down for the standard LMS. A similar result is obtained for the LDLMS algorithm.

3.1 System Model of The LDLMS Algorithm

As was shown in Chapter 2, the LLMS algorithm is derived based on minimizing the augmented instantaneous square error, $e(n)^2+\mu C(n)^T C(n)$. Then, the resulting updated equation is given by

$$C(n) = C(n-1) + \beta e(n)X(n) - \beta\mu C(n-1), \tag{3.1}$$

where $C(n)$ is the tap vector, $X(n)$ is the input vector, $e(n)$ is the error output, β and μ denote the step size and the leakage parameter, respectively.

In some situations, an inherent delay in the feedback error of equation (3.3) is a problem for the LLMS algorithm. This is referred to as leaky delayed LMS(LDLMS) algorithm and is given by

$$C(n) = C(n-1) + \beta e(n-D)X(n-D) - \beta\mu C(n-1) \tag{3.2}$$

In this adaptive version in which the delay is involved, the stability bound and convergence criterion of equation (3.2)must be examined anew. The analysis reduces to the previously derived ones for the LLMS algorithm.

The LDLMS algorithm in (3.2) may also be written in different forms as follows:

$$\begin{aligned} C(n) &= \gamma C(n-1) + \beta e(n-D)X(n-1) \\ C(n) &= C(n-1) + \beta e(n-D)X(n-D) - \alpha C(n-1) \end{aligned} \tag{3.3}$$

These algorithms are in fact the same but only the leakage parameter is differently stated. Form (3.3) of LDLMS is considered here, The estimated filter output is given by

$$y(n) = X^T(n)C(n-1) \tag{3.4}$$

and the estimated error is given by

$$e(n) = d(n) - y(n) \tag{3.5}$$

The input is considered to be stationary and is denoted by

$$X^T(n) = [x(n), x(n-1), \ldots, x(n-N+1)]$$

and the tap-weight vector is denoted by

$$C^T(n) = [c_0(n), c_1(n), \ldots, c_{N-1}(n)]$$

The quantities $d(n)$, β, and μ are already defined. Transpose operation for vectors and matrices is denoted by T. Now, we define the rotated coefficient error vector $\epsilon(n)$ and rotated input data vector $U(n)$ as

$$\begin{aligned} H &= V\Lambda V^T \\ W(n) &= V^T(C(n) - C_{opt}) = V^T\epsilon(n) \\ U(n) &= V^T X(n) \\ R_o &= V^T C_{opt} \end{aligned} \tag{3.6}$$

where V is an orthonormal matrix with columns which are the eigenvectors of H, and Λ is a diagonal matrix with the diagonal elements corresponding to the eigenvalues of H.

$$\Lambda = diag(\lambda_0, \lambda_1, \ldots, \lambda_{N-1})$$

The following fundamental assumptions will be used throughout the derivations.

1. $C(n)$ as given by (3.3) depends on $X(0), X(1), \ldots, X(n-D), \ldots, X(n-1)$, but it is independent of $X(n)$.

2. Each sample of the input vector is statistically independent of all the previous samples. i.e. $< X(n)X(k)^T >= 0$, $k = 0$ to $n-1$.

3. Each sample of the vector $X(n)$ is statistically independent of all previous samples of the desired vector, $d(k)$ i.e. $< X(n)d(k) >= 0$, $k = 0$ to $n-1$.

The above assumptions are used to facilitate mathematical treatment of the LDLMS algorithm. In fact, the fundamental assumptions are not strictly true. Since during adaptation, the error $e(n)$ and hence the tap-weight vector $c(n)$ behave in random manner. Therefore, the mean and variance of $C(n)$ and the mean squared-error MSE depend on time n. Nevertheless, experience has shown that when the step size is small enough, analysis of the adaptive algorithm behavior based on the fundamental assumptions agree closely with the empirical evaluation of the algorithm behavior [13, 27].

3.2 Coefficient Convergence of LMS, LLMS, DLMS, LDLMS Based on Root Locus Approach

The root locus is a technique widely employed in determining the stability of systems. In this section, we use this technique to study the stability behavior of LDLMS algorithm as well as LMS, LLMS, and DLMS algorithms. The recursion equation describing the coefficient update is obtained for all these four algorithms. This is then used to determine the poles of the recursion coefficient update equation of the system. The behavior of these poles as the step-size is increased is used to study the convergence and stability of these algorithms.

3.2.1 LMS Algorithm

The update equation of the LMS algorithm is rewritten

$$C(n) = C(n-1) + \beta e(n) X(n). \tag{3.7}$$

If we subtract C_{opt} from both sides of (3.7) we obtain the following:

$$\epsilon(n) = \epsilon(n-1) + \beta e(n) X(n) \tag{3.8}$$

The estimated error can be formulated as

$$e(n) = d(n) - X^T(n) C(n-1) \tag{3.9}$$

Thus, by substituting (3.9) into (3.8), taking the expectation along both sides, and defining $e_{opt}(n) = d(n) - X^T(n) C_{opt}$, we find,

$$< \epsilon(n) > = < \epsilon(n-1) > + \beta < e_{opt}(n) X(n) > - \beta < X(n) X^T(n) \epsilon(n-1) > . \tag{3.10}$$

Now using the fundamental assumptions and (3.6), the above equation reduces to

$$< W(n) > = < W(n-1) > - \beta \Lambda < W(n-1) >, \tag{3.11}$$

where $W(n)$ and Λ are as defined in (3.6). Let's denote the Z-transform of the i^{th} tap-error vector by $W_i(z)$. Then, taking the Z transform of both sides of the last equation, we obtain

$$zW_i(z) = W_i(z) - \beta\lambda_i W_i(z) + zW_i(0)$$

or

$$W_i(z) = \frac{zw_i(0)}{z - 1 + \beta\lambda_i} \tag{3.12}$$

The denominator of (3.12) is called the characteristic polynomial and is denoted by $F_i(z)$, i.e

$$F_i(z) = z - 1 + \beta\lambda_i. \tag{3.13}$$

As it can be seen from (3.13) , one real root exists which lies on the real axis of the root locus circle. Here, the root is in terms of step size and eigenvalues. Let's denote

$$z = e^{j\phi} \tag{3.14}$$

$$\Delta = \beta\lambda_i. \tag{3.15}$$

Now, (3.13) can be written as the following

$$F_i(z) = z - 1 + \Delta \tag{3.16}$$

By using (3.15) and (3.16), the following is obtained:

$$e^{j\phi} - 1 + \Delta_i = 0. \tag{3.17}$$

This can be expanded into a real part and an imaginary part as follows:

$$cos(\phi) - 1 + \Delta_i = 0 \tag{3.18}$$

$$sin(\phi) = 0 \tag{3.19}$$

The values of ϕ satisfying the latter equations define the points at which the root lies on the unit circle, i.e.

$$\phi = 0,\ \pi. \tag{3.20}$$

For $\Delta_i = 0$, this root lies on the unit circle at $\phi = 0$. As Δ_i increases, this root moves toward the negative axis of the unit circle. (see Figure 3.1).

The expression for the stability bound of the LMS algorithm can be deduced here by substituting (3.20) into (3.18) and it is given by

$$0 < \beta < \frac{2}{\lambda_i}. \tag{3.21}$$

This bound is identical to the one in [13] here obtained in a different way. A tighter upper bound can be found by considering the maximal eigenvalue, λ_{max} so we have,

$$0 < \beta < \frac{2}{\lambda_{max}}. \tag{3.22}$$

From (3.17), we see if $\lambda_i = 0$, the i^{th} mode of the adaptive filter would not be stable. Therefore, a pole is located on the unit circle at $\phi = 0$. Then, the LMS tap-weight filters may all not converge.

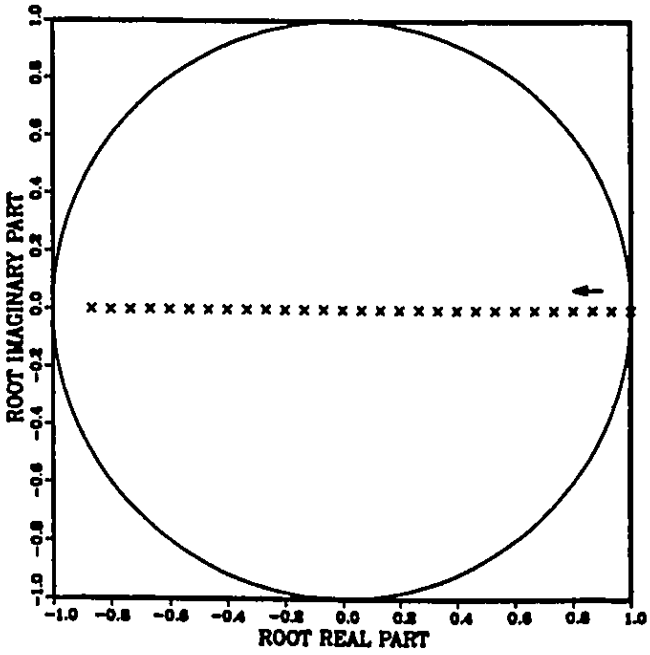


Figure 3.1: Root locus of the LMS characteristic polynomial. Variation of the real part vs the imaginary part of $F_i(z)$ root as Δ_i changes from zero to its upper bound.

3.2.2 LLMS Algorithm

The updated equation of the LLMS algorithm (2.25) is restated here,

$$\begin{aligned} C(n) &= (1-\beta\mu)C(n-1) + \beta e(n)X(n) \\ &= \gamma C(n-1) + \beta e(n)X(n), \end{aligned} \tag{3.23}$$

where $\gamma = 1 - \beta\mu$ and μ is the leakage parameter. This notation is preferred to simplify the mathematical analysis of LLMS stability bounds. If we subtract C_{opt} from both sides of (3.23) and using (3.9) along with the fundamental assumptions, we obtain

$$< W(n) >= \gamma < W(n-1) > -\beta\Lambda < W(n-1) > +R_o(1-\gamma) \tag{3.24}$$

where R_o is the rotated optimum weight defined in (3.6). This difference equation consists of two parts. The two right hand terms are part of the natural response of the system while in the second part, the last term $R_o(1-\gamma)$ constitutes the forced response on the system. The behavior of the system can be well understood by studying its natural response or the homogeneous part of (3.24). The remaining terms contribute to a bias on the true optimum weights C_{opt}. Thus, the homogeneous part of equation (3.24) is given by

$$< W(n) >= \gamma < W(n-1) > -\beta\Lambda < W(n-1) > \tag{3.25}$$

Taking the Z-transform the i^{th} tap coefficient error, we found

$$W_i(z) = \frac{zw_i(0)}{z - \gamma + \beta\lambda_i} \tag{3.26}$$

As before, the denominator of the above equation is described by

$$F_i(z) = z - \gamma + \Delta_i. \tag{3.27}$$

This equation is solved for different values of the leakage factor γ. Whenever $\Delta_i = 0$, we always have $z = \gamma$. This root corresponds to a bias term that exists in the nonexciting mode of adaptive filter. This is the situation where one or more eigenvalues of the auto-correlation input matrix is zero. As Δ_i increases from zero, the root on the positive axis moves toward the negative axis. In this case, a pole on the unit circle is pulled inside the

unit circle by amount of $1-\gamma$. Thus, leakage ensures stability of the adaptive filter by preventing poles from drifting outside the unit circle. Figures 3.2- 3.3 show the root locus of the LLMS for different leakage values.

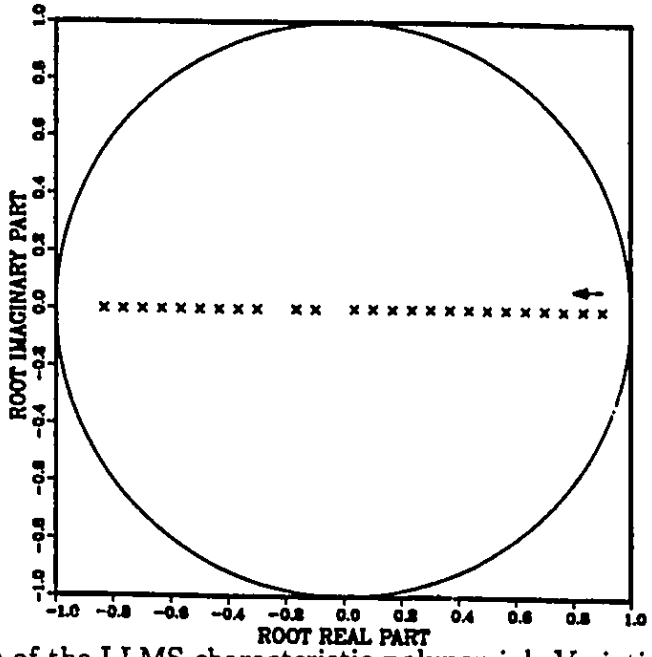


Figure 3.2: Root locus of the LLMS characteristic polynomial: Variation of the real part vs the imaginary part of $F_i(z)$ root as Δ_i changes from zero to its upper bound:$\gamma = 0.9$.

3.2.3 DLMS Algorithm

The DLMS algorithm was reviewed in Chapter 2. Its rotated tap-weight error vector can be given by

$$< W(n) >=< W(n-1) > -\beta\Lambda < W(n-D-1) > . \tag{3.28}$$

In this case, the DLMS algorithm has the Z transform of the tap-weight error [6]

$$W_i(z) = \frac{z w_i^{D+1}(0)}{z^{D+1} - z^D + \Delta_i}, \tag{3.29}$$

and its characteristic equation is given by

$$F_i(z) = z^{D+1} - z^D + \Delta_i. \tag{3.30}$$

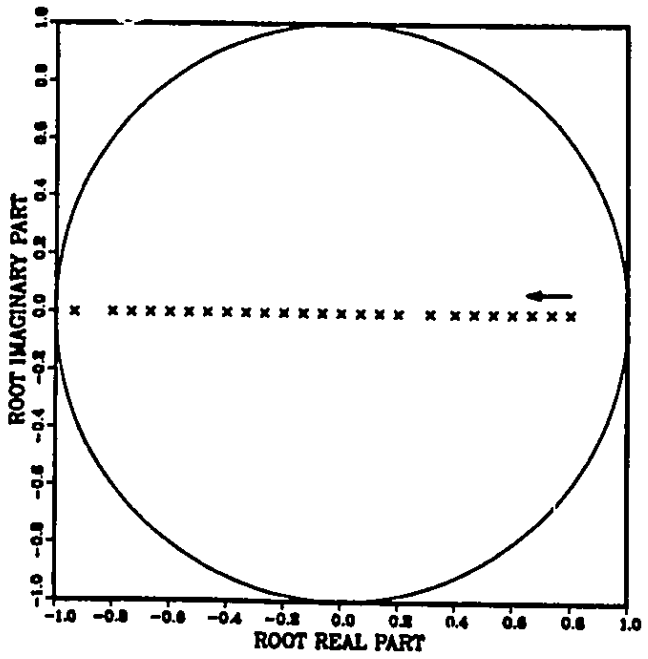


Figure 3.3: Root locus of the LLMS characteristic polynomial: Variation of the real part vs the imaginary part of $F_i(z)$ root as Δ_i changes from zero to its upper bound:$\gamma = 0.8$.

The system is stable if all the roots of $F_i(z)$ lie inside the unit circle. Then, bounds on Δ_i corresponding to the positions of these roots which lie inside the unit have to be found. Employing numerical methods to solve the above polynomial, Figures 3.4 and 3.5 show that for Δ_i=0, D roots lie at the origin of the root locus and one root lies on the unit circle. As Δ_i increases, the root at $z = 1$ and one of the other roots move toward each other on the real axis to meet at $z = \frac{D}{D+1}$ and then they break away to form a complex conjugate pair, which then cross the unit circle at $\phi = \frac{\pi}{2D+1}$. However, the remaining roots move radially to the real axis from the origin of the system axis.

We note that if λ_i is zero for a fixed value of β, this mode of the adaptive filter will not converge and may be unstable. The stability bound on the mean coefficient is given in [6] by

$$0 < \beta < \frac{2}{\lambda_{max}} sin\left(\frac{\pi}{2(2D+1)}\right). \tag{3.31}$$

This bound guarantees the stability of the algorithm, although the tap-weight may diverge if one or more eigenvalues are zero.

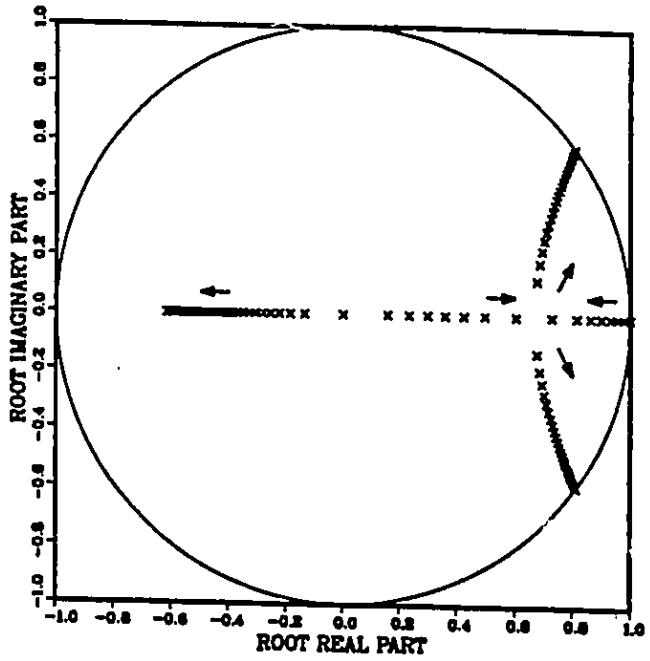


Figure 3.4: Root locus of the DLMS characteristic polynomial: Variation of the real part vs the imaginary part of $F_i(z)$ roots as Δ_i changes from zero to its upper bound: $D = 2$.

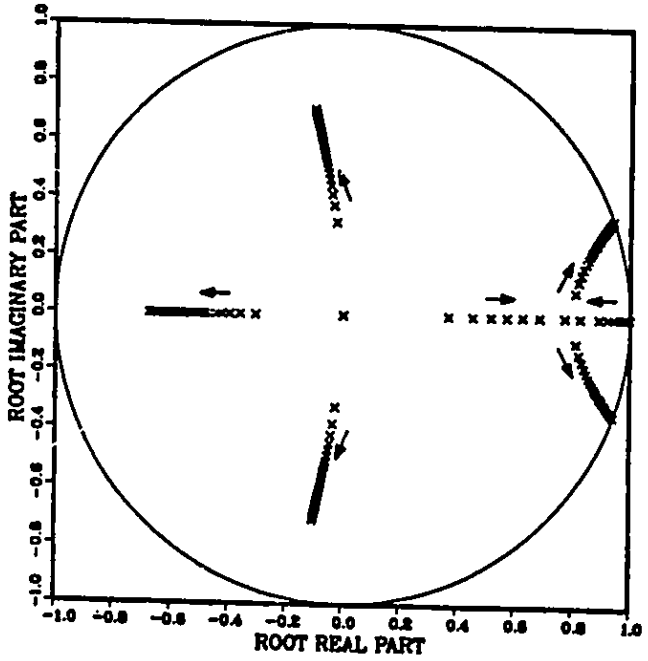


Figure 3.5: Root locus of the DLMS characteristic polynomial: Variation of the real part vs the imaginary part of $F_i(z)$ roots as Δ_i changes from zero to its upper bound: $D = 4$.

3.2.4 LDLMS Algorithm

The LDLMS algorithm is rewritten in terms of the rotated tap-weight error vector here

$$< W(n) >= \gamma < W(n-1) > -\beta\Lambda < W(n-D-1) > +R_o(1-\gamma), \tag{3.32}$$

where R_o is defined as before. The above equation is identical to the LLMS algorithm except for the presence of the delay that shows up in the correction term of the update equation. Considering only the homogeneous part of (3.32), the Z-transform of the i^{th} tap error vector is given by

$$W_i(z) = \frac{zw_i^{D+1}(0)}{z^{D+1} - \gamma z^D + \Delta_i}, \tag{3.33}$$

and its characteristic function is described by

$$F_i(z) = z^{D+1} - \gamma z^D + \Delta_i. \tag{3.34}$$

The last equation is similar to $F_i(z)$ of the DLMS but exact expression of the upper stability bound of the LDLMS cannot be found. Because of the presence of the leakage coefficient, numerical methods are to be utilized to solve for $F_i(z)$. From Figures 3.6-Figure 3.9, we note that the root at $z = 1$ in DLMS case is now at $z = \gamma$ instead. This is an important feature of the leakage which tries to push the root (pole) inside the unit circle.

Now, when Δ_i gets larger, the two roots that move toward each other split to form a complex conjugate pair much quicker than in the case of DLMS, depending on how much leakage is introduced in the algorithm. In addition, for the DLMS algorithm, if $\Delta_i = 0$ either for very small or negligible eigenvalues λ_i, this i^{th} mode of the adaptive filter will not be stable. However, with presence of leakage γ, null eigenvalues do not destabilize the system. This shows the robustness of the LDLMS algorithm compared to the LMS algorithm with delayed coefficients.

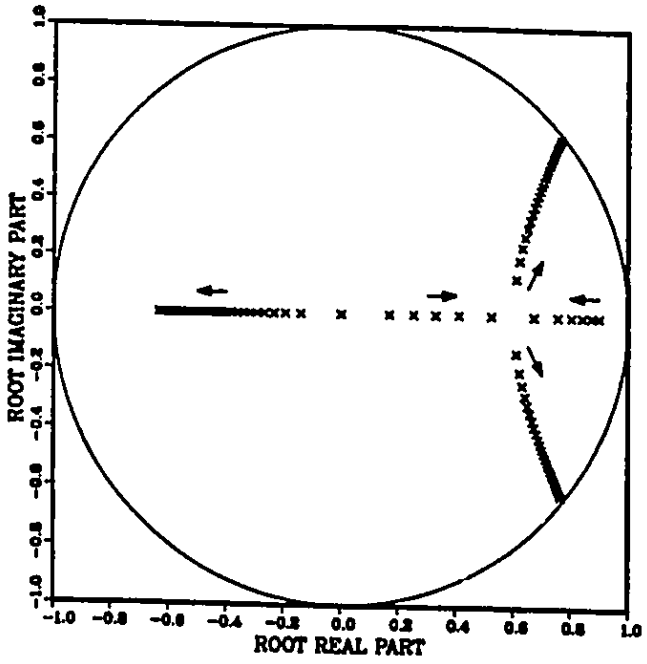


Figure 3.6: Root locus of the LDLMS characteristic polynomial: Variation of the real part vs the imaginary part of $F_i(z)$ roots as Δ_i changes from zero to its upper bound: $D = 2, \gamma = 0.9$.

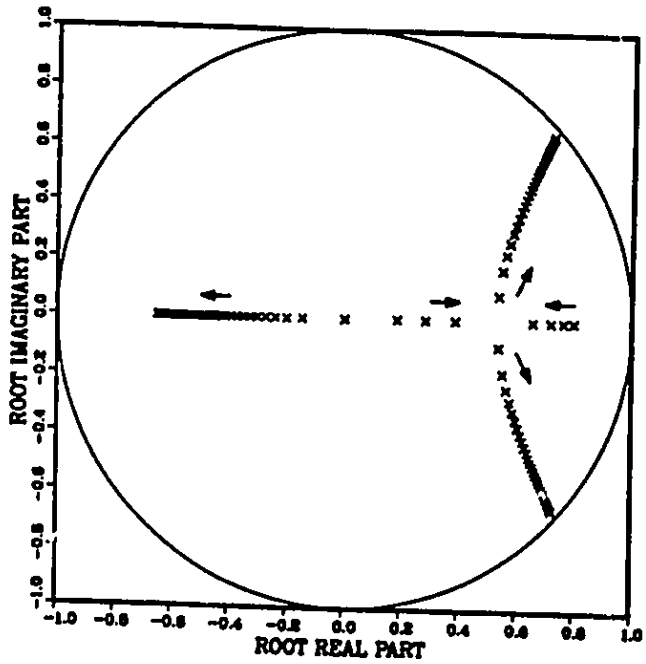


Figure 3.7: Root locus of the LDLMS characteristic polynomial: Variation of the real part vs the imaginary part of $F_i(z)$ roots as Δ_i changes from zero to its upper bound: $D = 2, \gamma = 0.8$.

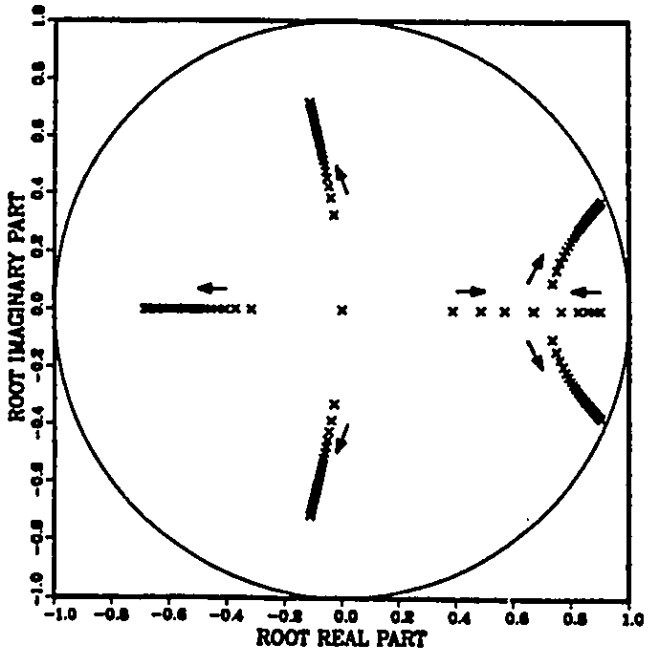


Figure 3.8: Root locus of the LDLMS characteristic polynomial: Variation of the real part vs the imaginary part of $F_i(z)$ roots as Δ_i changes from zero to its upper bound: $D = 4, \gamma = 0.9$.

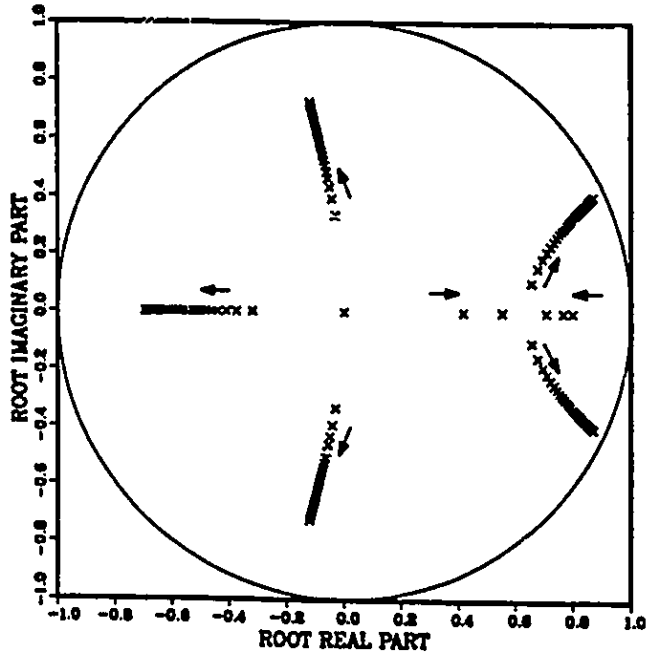


Figure 3.9: Root locus of the LDLMS characteristic polynomial: Variation of the real part vs the imaginary part of $F_i(z)$ roots as Δ_i changes from zero to its upper bound: $D = 4, \gamma = 0.8$.

3.3 Error Convergence of the Leaky Delayed LMS Algorithm

Here, the residual square error of the leaky delayed LMS (LDLMS) is examined by using the following instantaneous modified cost function:

$$J(n) = J_{min} + J_{ex}(n) \tag{3.35}$$

where J_{min} is the minimum MSE of the LMS algorithm. An expression for the steady-state excess MSE $J_{ex}(\infty)$ is computed in the following manner. The excess MSE of the LDLMS algorithm is given by

$$J_{ex}(n) =< \epsilon(n)^T H \epsilon(n) > \tag{3.36}$$

where

$$\epsilon(n) = C_n - C_{opt} \tag{3.37}$$

is the tap-error vector.

If we express (3.3) in terms of $\epsilon(n)$ and then use the transformation techniques in equation (3.6) along with the fundamental assumptions, the following equations are obtained,

$$W_n = (1 - \beta\mu)W_{n-1} + \beta e_{opt}(n - D)U_{n-D} - \beta U_{n-D}U_{n-D}^T W_{n-D-1} - \beta\mu R_o \tag{3.38}$$

$$J_{ex}(n-1) =< W_{n-1}^T \Lambda W_{n-1} >, \tag{3.39}$$

where $e_{opt}(n - D)$ is the instantaneous error when the adaptive tap-weights are at their optimum settings and it is given by:

$$e_{opt}(n - D) = d(n - D) - y(n - D), \tag{3.40}$$

and $R_o = V^T C_o$ and V is the eigenvector matrix of the input autocorrelation matrix H. $U(n)$ i.. the rotated input vector.

Now, by substituting (3.38) into (3.39), we obtain

$$J_{ex}(n) \;=\; < W_n^T \Lambda W_n >$$

$$
\begin{aligned}
= \; & (1-\beta\mu)^2 < W_{n-1}^T \Lambda W_{n-1} > + \beta^2 < e_{opt}^2(n-D) U_{n-D}^T \Lambda U_{n-D} > \\
& + \beta^2 < W_{n-D-1} U_{n-D} U_{n-D}^T \Lambda U_{n-D} U_{n-D}^T W_{n-D-1} > \\
& + \beta^2 \mu^2 R_o^T \Lambda R_o - (1-\beta\mu)\beta < W_{n-D-1}^T U_{n-D} U_{n-D}^T \Lambda W_{n-1} > \\
& + \beta^2 \mu < W_{n-D-1}^T U_{n-D} U_{n-D}^T \Lambda R > - \beta\mu(1-\beta\mu) < R_o^T \Lambda W_{n-1} > \\
& + \beta(1-\beta\mu) < e_{opt}(n-D) U^T \Lambda W_{n-1} > - \beta^2 \mu < e_{opt}(n-D) U_{n-D}^T \Lambda R_o > \\
& - \beta^2 < e_{opt}(n-D) U_{n-D}^T \Lambda U_{n-D} U_{n-D}^T W_{n-D-1} > \\
& + \beta^2 \mu^2 R_o^T \Lambda R_o - (1-\beta\mu)\beta < W_{n-1}^T U_{n-D} U_{n-D}^T \Lambda W_{n-D-1} > \\
& - \beta\mu(1-\beta\mu) < W_{n-1}^T \Lambda R_o^T > \\
& + \beta(1-\beta\mu) < W_{n-1}^T \Lambda e_{opt}(n-D) U_{n-D} > - \beta^2 \mu < R_o^T \Lambda e_{opt}(n-D) U_{n-D} > \\
& - \beta^2 < W_{n-D-1}^T U_{n-D} U_{n-D}^T \Lambda e_{opt}(n-D) U_{n-D} > \qquad (3.41)
\end{aligned}
$$

Note that with the assumptions that $X(n)$, $\epsilon(n)$, and $e_{opt}(n)$ are statistically independent, $U(n)$, $W(n)$, and $e_{opt}(n)$ are also statistically independent. At the optimum tap settings, the error signal is uncorrelated with the current input vector. Thus, (3.41) can be simplified as follows:

1. The first term of the above equation is obviously equal to

$$< W_{n-1}^T \Lambda W_{n-1} >= J_{ex}(n-1) \qquad (3.42)$$

2. By using the fundamental assumptions on the inputs, the 2^{nd} term in (3.41) can be reduced to the following:

$$
\begin{aligned}
& < e_{opt}^2(n-D) U_n^T \Lambda U_n >=< e_{opt}^2(n-D) >< \sum_{i=0}^{N-1} \lambda_i u_i^2(n-D) > \\
& = J_{min} tr(H^2) = J_{min} \sum_{i=0}^{N-1} \lambda_i^2 \\
& = J_{min} N \lambda_{rms}^2 = J_{min} \alpha N \sigma_x^4 \qquad (3.43)
\end{aligned}
$$

where $tr\{H^2\}$ is the sum of the diagonal elements of H^2 and λ_{rms} is the root mean-squared eigenvalue given by

$$\lambda_{rms}^2 = \frac{1}{N} \sum_{i=0}^{N-1} \lambda_i^2 \qquad (3.44)$$

3. For the 3^{rd} term, it can be simplified as follows:

$$< U_{n-D}U_{n-D}^T\Lambda U_{n-D}U_{n-D}^T >=< U_{n-D}\sum_{i=0}^{N-1}\lambda_i u_i^2(n-D)U_{n-D}^T > \tag{3.45}$$

For simplicity, a single element of this matrix is considered. So, it follows

$$\begin{aligned}
& < \sum_{i=0}^{N-1} u_k(n-D)u_m(n-D)\lambda_i u_i^2(n-D) > \\
= \;& \delta_{km}\left[\sum_{i=0,i\neq k}^{N-1}\lambda_i < u_k^2(n-D) < u_i^2(n-D) > +\lambda_k < u_k^4(n-D) >\right] \\
= \;& \delta_{km}\left[\sum_{i=0,i\neq k}^{N-1}\lambda_i < u_k^2(n-D) >< u_i^2(n-D) > +\lambda_k < u_k^4(n-D) >\right] \\
= \;& \delta_{km}\left[\sum_{i=0}^{N-1}\lambda_i < u_k^2(n-D) >< u_i^2(n-D) > +\lambda_k < u_k^4(n-D) > -\lambda_k < u_k^2(n-D) >^2\right] \\
= \;& \delta_{km}\left[\sum_{i=0}^{N-1}\lambda_i < u_k^2(n-D) >< u_i^2(n-D) > +(\nu_x(k)-1)\lambda_k < u_k^2(n-D) >^2\right] \\
= \;& \delta_{km}\left[(\sum_{i=0}^{N-1}\lambda_i^2)\lambda_k + (\nu_x(k)-1)\lambda_k^3\right]
\end{aligned} \tag{3.46}$$

where δ_{km} is the Kronecker delta function and $\nu_x(k)$ is the kurtosis which is defined by $\nu_x(k) = \frac{<u_k^4>}{<u_k^2>^2}$. To make the mathematical analysis tractable, $\nu_x(k)$ is assumed constant for different values of k which is a valid assumption in most practical cases [27]. Now we express (3.46) in matrix form as follows:

$$\begin{aligned}
& < W_{n-D-1}^T U_{n-D}U_{n-D}^T\Lambda U_{n-D}U_{n-D}^T W_{n-D-1} > \\
= \;& N\lambda_{rms}^2 < W_{n-D-1}^T\Lambda W_{n-D-1} > +(\nu_x-1) < W_{n-D-1}^T\Lambda^3 W_{n-D-1} > .
\end{aligned} \tag{3.47}$$

The first term on the right side in (3.47) is equal to $N\lambda_{rms}^2 J_{ex}(n-D-1)$. However, the second term can be approximated as follows:

$$\begin{aligned}
& (\nu_x-1) < W_{n-D-1}^T\Lambda^3 W_{n-D-1} > \\
& = (\nu_x-1) < W_{n-D-1}^T\Lambda^2\Lambda W_{n-D-1} > \\
& \approx \lambda_{rms}^2(\nu_x-1) < W_{n-D-1}^T\Lambda W_{n-D-1} > \\
& = (\nu_x-1)\lambda_{rms}^2 J_{ex}(n-D-1).
\end{aligned} \tag{3.48}$$

This approximation is valid only when the eigenvalues of Λ^2 are uniform. Neverthcless, experiments showed that this condition will not affect the results when it is violated [4].

Thus, adding up all these terms, we find

$$\begin{aligned} &< W_{n-D-1}^T U_{n-D} U_{n-D}^T \Lambda U_{n-D} U_{n-D}^T W_{n-D-1} > \\ &= N\lambda_{rms}^2 J_{ex}(n-D-1) + \lambda_{rms}^2(\nu_x - 1) J_{ex}(n-D-1) \\ &= \alpha M \sigma_x^4 J_{ex}(n-D-1) \end{aligned} \tag{3.49}$$

where $M = N + \nu_x - 1$ and $\alpha = \frac{\lambda_{rms}^2}{\lambda_{avg}}$ and the average eigenvalue λ_{avg} is given by

$$\lambda_{avg} = \frac{1}{N} \sum_{i=0}^{N-1} \lambda_i. \tag{3.50}$$

4. The fourth term of equation (3.41) is a constant and it can be simplified to

$$\beta^2 \mu^2 R_o^T \Lambda R_o = \beta^2 \mu^2 P^T H^{-1} P. \tag{3.51}$$

The above term is very small because of the product term $(\beta\mu)^2$.

5. The fifth term of the main equation can also be reduced as

$$\begin{aligned} < W_{n-D-1}^T U_{n-D} U_{n-D}^T \Lambda W_{n-1} > &= < W_{n-D-1}^T \Lambda^2 W_{n-1} > \\ &= \sum_{i=0}^{N-1} \lambda_i^2 < w_i(n-1) w_i(n-D-1) > . \end{aligned} \tag{3.52}$$

This term is upper and lower bounded by

$$\lambda_{min} < W_{n-1} \Lambda W_{n-D-1} > \leq < W_{n-D-1}^T \Lambda^2 W_{n-1} > \leq \lambda_{max} < W_{n-1} \Lambda W_{n-D-1} > \tag{3.53}$$

where λ_{min} and λ_{max} are the smallest and biggest eigenvalues of Λ. However, in practice, usually no information is available about these two quantities. The lower bound of (3.53) is loose because the smallest eigenvalue λ_{min} can drastically reduce the magnitude of $< W_{n-1} \Lambda W_{n-D-1} >$. However, if we assume that $< w_i(n-1) w_i(n-D-1) >$ is relatively uniform for all i, then a reasonable approximation on this bound can be made by

$$< W_{n-D-1}^T U_{n-D} U_{n-D}^T \Lambda W_{n-1} >$$

$$
\begin{aligned}
&= < W_{n-D-1}^T \Lambda^2 W_{n-1} > \\
&\approx \lambda_{avg} < W_{n-D-1}^T \Lambda W_{n-1} > \\
&= \lambda_{avg} a_{D,D}(n-1) \qquad (3.54)
\end{aligned}
$$

where $a_{D,D}(n-1)$ is denoted by the following:

$$
a_{D,D}(n-1) = < W_{n-D-1}^T \Lambda W_{n-1} > . \qquad (3.55)
$$

6. Furthermore, the 8^{th}, 9^{th}, and 10^{th} terms are zero since $< e_{opt}(n-D) >= 0$.

7. Identical terms of the main equation can be simplified in the same way as before.

Finally, The overall equation (3.41) now can be written as

$$
\begin{aligned}
J_{ex}(n) \approx\ & (1-\beta\mu)^2 J_{ex}(n-1) + \beta^2 \alpha J_{min} N \sigma_x^4 + \alpha M \sigma_x^4 \beta^2 J_{ex}(n-D-1) \\
& + \beta^2 \mu^2 P^T H^{-1} P - 2\beta(1-\beta\mu)\sigma_x^2 a_{D,D}(n-1) \\
& + 2\beta^2 \mu \sigma^4 < R_o^T W_{n-D-1} > -2\beta\mu(1-\beta\mu)\sigma_x^2 < R_o^T W_{n-1} > \qquad (3.56)
\end{aligned}
$$

3.4 Steady State Excess MSE of LDLMS Algorithm

In order to solve (3.56), the relationship between $a_{D,D}(n-1)$ and $J_{ex}(n)$ has to be found. The derivation of equation (3.56) as function of the excess MSE is done in Appendix A. After substitution of this term in equation (3.56), we have

$$
\begin{aligned}
J_{ex}(n) =\ & (1-\beta\mu)^2 J_{ex}(n-1) + \beta^2 \alpha J_{min} N \sigma_x^4 + \alpha M \sigma_x^4 \beta^2 J_{ex}(n-D-1) \\
& + \beta^2 \mu^2 P^T H^{-1} P - 2\beta(1-\beta\mu)\sigma_x^2 J_{ex}(n-D-1) \\
& + 2\beta^2 \mu \sigma_x^4 < R_o^T W_{n-D-1} > -2\beta\mu(1-\beta\mu)\sigma_x^2 < R_o^T W_{n-1} > \\
& + 2\beta^2(1-\beta\mu)\sigma_x^4 \sum_{j=1}^{D} J_{ex}(n-D-j-1) \\
& - 2\beta^3(1-\beta\mu)\sigma_x^6 \sum_{j=1}^{D} \sum_{i=D-j+1}^{D} J_{ex}(n-D-j-i-1) \\
& + 2\beta^4(1-\beta\mu)\sigma_x^8 \sum_{j=1}^{D} \sum_{i=D-j+1}^{D} \sum_{s=D-i+1}^{D} J_{ex}(n-D-j-i-s-1) \\
& - 2\beta^5(1-\beta\mu)\sigma_x^{10} \sum_{j=1}^{D} \sum_{i=D-j+1}^{D} \sum_{s=D-i+1}^{D} \sum_{r=D-s-1}^{D} J_{ex}(n-D-j-i-s-r-1)
\end{aligned}
$$

$$-\beta^2\mu\sigma_x^2 D\ldots \tag{3.57}$$

Considering the steady state excess MSE at time n when the algorithm converges, we can assume that at the steady state,

$$J_{ex}(n) \approx J_{ex}(n-1) \approx \cdots \approx J_{ex}(\infty). \tag{3.58}$$

Solving for $J_{ex}(\infty)$, An approximation result of the steady state excess MSE is obtained in the following:

$$J_{ex}(\infty) \cong \frac{\beta J_{min}\alpha N\sigma_x^4 + \beta\mu^2 P^T H^{-1} P + 2\beta\mu\sigma_x^2 P^T\epsilon}{2(\sigma_x^2+\mu) - \beta(2\mu\sigma_x^2 + \alpha M\sigma_x^4 + \mu^2 + 2\sigma_x^4 D) + \sigma_x^4\beta^2(2\mu D + \sigma_x^2 D(D+1))} \cdot \frac{1}{-\beta^3\sigma_x^6 D(D+1)(\mu + \frac{1}{3}(2D+1)\sigma_x^2) + \beta^4\sigma_x^8 D(D+1)\left[(2D+1)\mu + \frac{(D+2)(3D+1)}{6}\right]\ldots} \tag{3.59}$$

where ϵ is the tap-weight error vector when the algorithm converges. This steady state excess MSE depends on the leakage parameter. A bias term on the true optimum weights appears in the numerator. This accounts for a tradeoff that has to exist between damping the undriven adaptive tap-weights and increasing the residual output error power.

3.5 Stability Bounds of LDLMS Algorithm

The denominator of equation (3.59) has an infinite number of terms. However, in the region of stability, the step size is normally small. Hence, terms of higher power order decay quickly. The stability bound on the step size is found by determining β that makes this denominator equal to zero. i.e. solving for the roots of the following polynomial:

$$\begin{aligned} S(\beta) &= 2(\sigma_x^2+\mu) - \beta(2\mu\sigma_x^2 + \alpha M\sigma_x^4 + \mu^2 + 2\sigma_x^4 D) + \sigma_x^4\beta^2(2\mu D + \sigma_x^2 D(D+1)) \\ &\quad -\beta^3\sigma_x^6 D(D+1)(\mu + \frac{1}{3}(2D+1)\sigma_x^2) \\ &\quad +\beta^4\sigma_x^8 D(D+1)\left[(2D+1)\mu + \frac{(D+2)(3D+1)}{6}\right] + \ldots \end{aligned}$$

$$\approx 2(\sigma_x^2+\mu)-\beta(2\mu\sigma_x^2+\alpha M\sigma_x^4+\mu^2+2\sigma_x^4D)+\sigma_x^4\beta^2(2\mu D+\sigma_x^2D(D+1))$$
$$-\beta^3\sigma_x^6D(D+1)(\mu+\frac{1}{3}(2D+1)\sigma_x^2)$$
$$+\beta^4\sigma_x^8D(D+1)\left[(2D+1)\mu+\frac{(D+2)(3D+1)}{6}\right] \quad (3.60)$$

As β increases from zero to a value β_{max}, $S(\beta)$ decreases monotonically from S(0)=$2(\sigma_x^2+\mu)$ to $S(\beta_{max})=0$ where $J_{ex}(\infty)$ at β_{max} blows up.

For applications where $\beta D << 1$, $S(\beta)$ can be approximated as

$$S(\beta)=2(\sigma_x^2+\mu)-\beta(2\mu\sigma_x^2+\alpha M\sigma_x^4+\mu^2+2\sigma_x^4D). \quad (3.61)$$

In this case, the upper bound on the step size can be easily found

$$\beta_{max}=\frac{2(\sigma_x^2+\mu)}{2\mu\sigma_x^2+\alpha M\sigma_x^4+\mu^2+2\sigma_x^4D} \quad (3.62)$$

and the steady state excess MSE is then given by

$$J_{ex}(\infty) \cong \frac{\beta J_{min}\alpha N\sigma_x^4+\beta\mu^2P^TH^{-1}P+2\beta\mu\sigma_x^2P^T\epsilon}{2(\sigma_x^2+\mu)-\beta(2\mu\sigma_x^2+\alpha M\sigma_x^4+\mu^2+2\sigma_x^4D)} \quad (3.63)$$

However, for moderately large values of D, the second order term has to be retained. i.e.,

$$S(\beta)=2(\sigma_x^2+\mu)-\beta(2\mu\sigma_x^2+\alpha M\sigma_x^4+\mu^2+2\sigma_x^4D)$$
$$+\sigma_x^4\beta^2(2\mu D+\sigma_x^2D(D+1)). \quad (3.64)$$

The other terms with higher power order are neglected. So, the maximum step size which constitutes the upper bound is now given by:

$$\beta_{max}=\frac{A_2-\sqrt{A_2^2-4A_1A_3}}{2A_1} \quad (3.65)$$

where

$$A_1 = 2\mu D+\sigma_x^2D(D+1)$$
$$A_2 = 2\mu\sigma_x^2+\alpha M\sigma_x^4+\mu^2+2\sigma_x^4D$$
$$A_3 = 2(\sigma_x^2+\mu), \quad (3.66)$$

and equation (3.59) can be approximated to

$$J_{ex}(\infty) \cong \frac{\beta J_{min}\alpha N\sigma_x^4 + \beta\mu^2 P^T H^{-1} P + 2\beta\mu\sigma_x^2 P^T \epsilon}{2(\sigma_x^2+\mu) - \beta(2\mu\sigma_x^2 + \alpha M\sigma_x^4 + \mu^2 + 2\sigma_x^4 D) + \sigma_x^4\beta^2(2\mu D + \sigma_x^2 D(D+1))} \frac{1}{-\beta^3\sigma_x^6 D(D+1)(\mu + \frac{1}{3}(2D+1)\sigma_x^2)}. \tag{3.67}$$

For larger values of βD, if the term with 3^{rd} power order has to be included, numerical methods have to be employed to solve the polynomial as in equation (3.60). From Figures 3.10- 3.17 which show the approximated stability bounds for different filter lengths N, we deduce the following:

- As a rule of thumb, the 1^{st}, 2^{nd}, 3^{rd}, and 4^{th} power order approximation in the above figures can be selected to determine step size bound when $\frac{D}{\alpha M} \leq 0.5$. On the other hand, when $0.5 \leq \frac{D}{\alpha M} < 1$, either 3^{rd} or 4^{th} approximation has to be chosen.

- As $\frac{D}{\alpha M}$ ratio exceeds one, all approximations fail to get an upper bound for the step size. Thus, a higher order power approximation has to be found. Nevertheless, for practical purposes, $\frac{D}{\alpha M} \ll 1$, otherwise degradation of the DLMS algorithm begins to appear.

- Looking at Figures 3.10- 3.17, we can see that increasing the order of approximation to the 5^{th} and 6^{th} power order and so on, approximation curves will converge afterwards i.e. 3^{rd} and 4^{th} are between 1^{st} and 2^{nd} ;5^{th} and 6^{th} will be between 3^{rd} and 4^{th}.

The effect of the leakage on the approximated stability bound can be summarized as follows:

- The approximations made for the derivation of the steady state excess MSE and the stability bounds of the LDLMS algorithm are only accurate for smaller leakage values. In fact, in most applications of leakage algorithms, a small value of μ is

employed. Larger values of leakage introduce a degradation on the performance of the algorithm, which may not be acceptable.

- For n ... ero values of leakage, the step size is slightly larger than in the case of zero leakage μ. Therefore, even with this increase in the stability bound, the MSE of the LDLMS algorithm is stable.

Figures 3.10- 3.17 show that all the approximations of the stability bound of the LDLMS are not accurate for higher delays. This comes from the approximations made on the derivation of the system equation (3.57). We note that the bounds of fourth order power term and the third order power term are better than the first and second order ones. The latters are of acceptable accuracy, good only for small value of the ratio $\frac{D}{N}$.

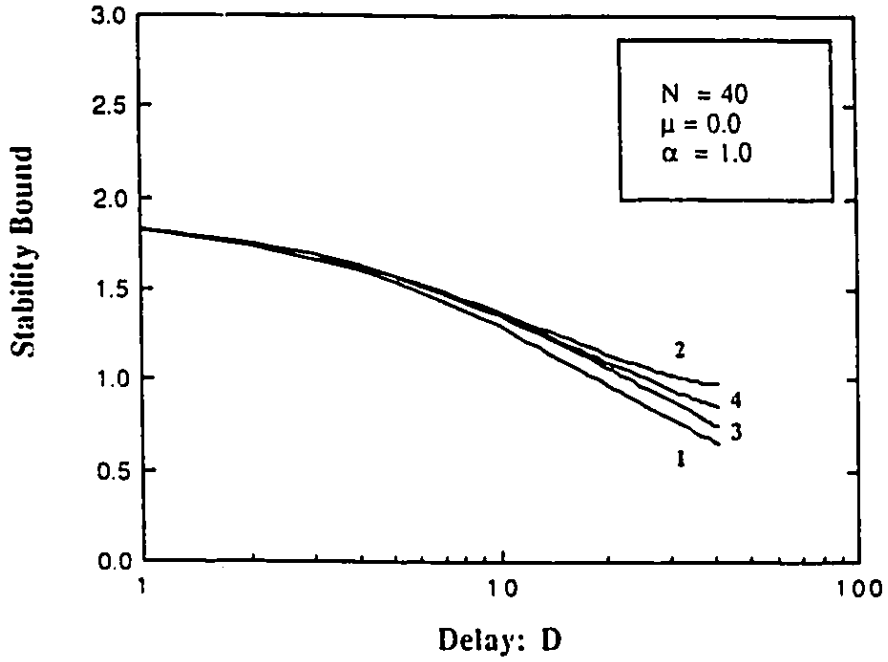


Figure 3.10: Stability bounds ($N\beta\sigma_x^2$) vs Delay with $\mu = 0$ and $N = 40$: Accuracy approximation of the LDLMS stability bounds on the step size.

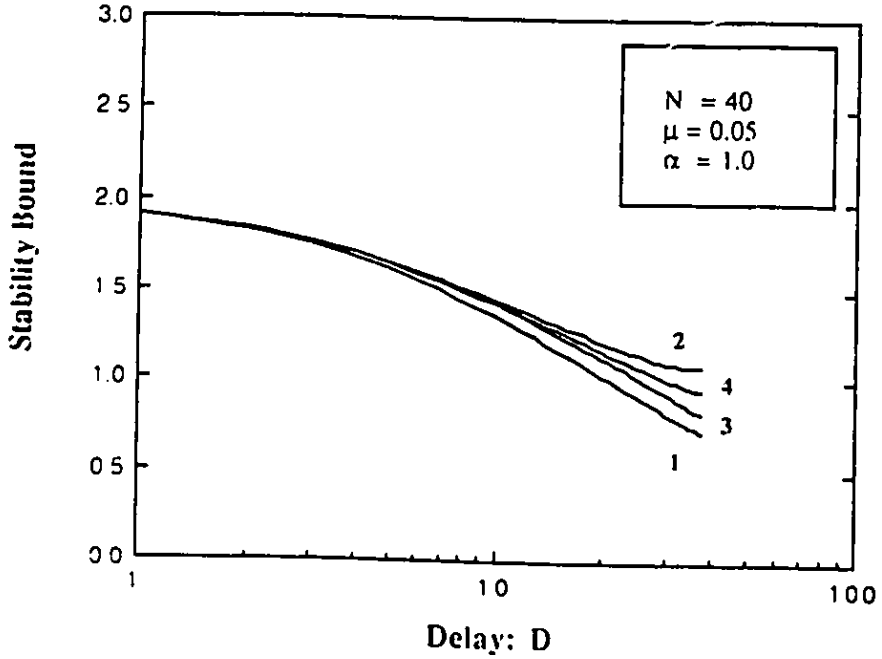


Figure 3.11: Stability bounds ($N\beta\sigma_x^2$) vs Delay with $\mu = 0.05$ and $N = 40$: Accuracy approximation of the LDLMS stability bounds on the step size.

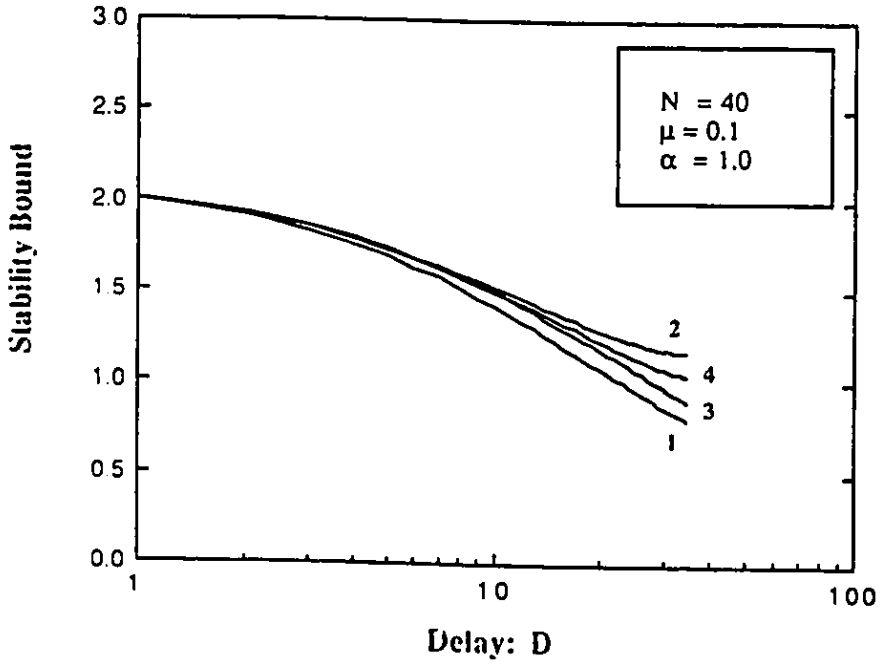


Figure 3.12: Stability bounds ($N\beta\sigma_x^2$) vs Delay with $\mu = 0.1$ and $N = 40$: Accuracy approximation of the LDLMS stability bounds on the step size.

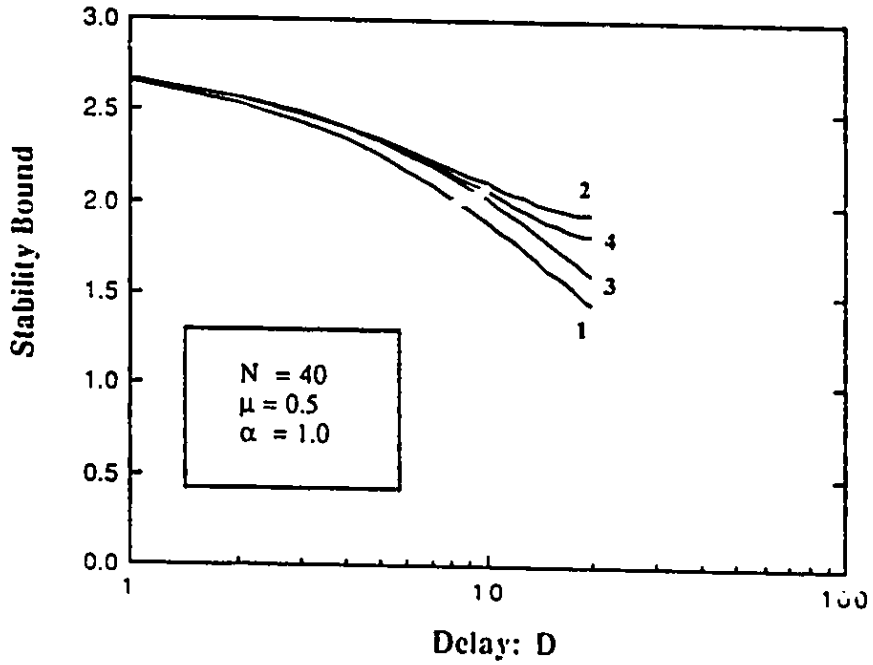


Figure 3.13: Stability bounds ($N\beta\sigma_x^2$) vs Delay with $\mu = 0.5$ and $N = 40$: Accuracy approximation of the LDLMS stability bounds on the step size.

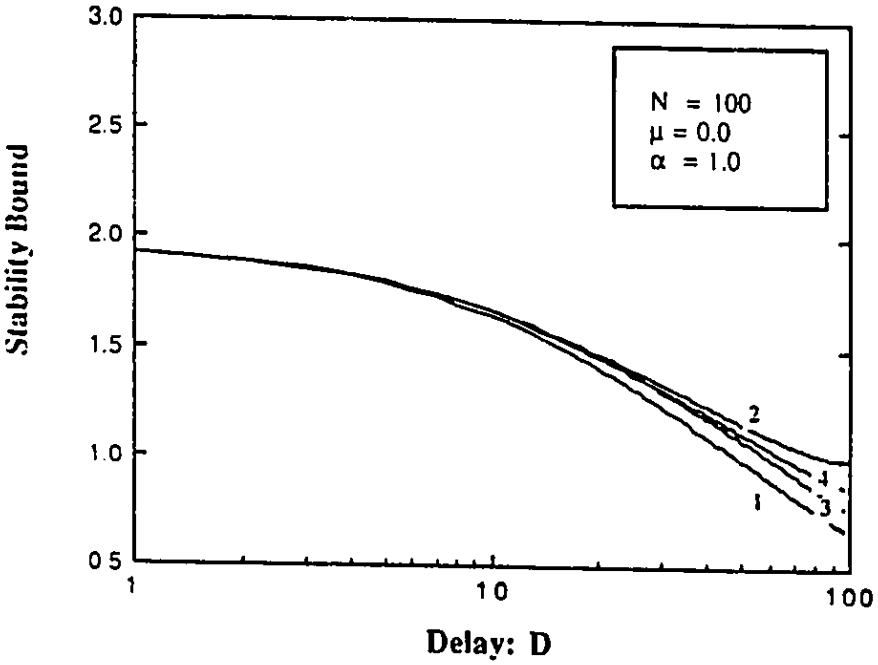


Figure 3.14: Stability bounds ($N\beta\sigma_x^2$) vs Delay with $\mu = 0.0$ and $N = 100$: Accuracy approximation of the LDLMS stability bounds on the step size.

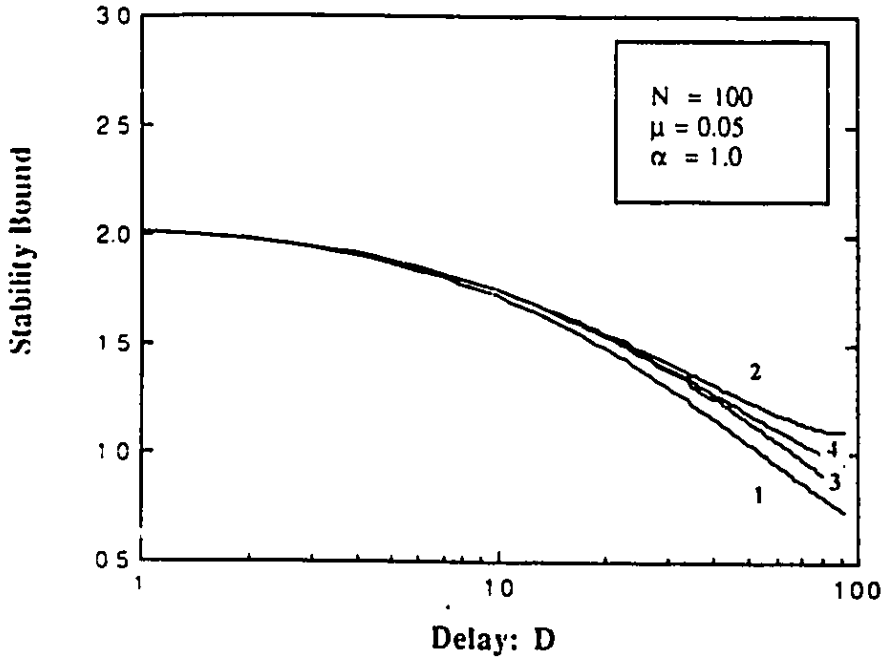


Figure 3.15: Stability bounds ($N\beta\sigma_x^2$) vs Delay with $\mu = 0.05$ and $N = 100$: Accuracy approximation of the LDLMS stability bounds on the step size.

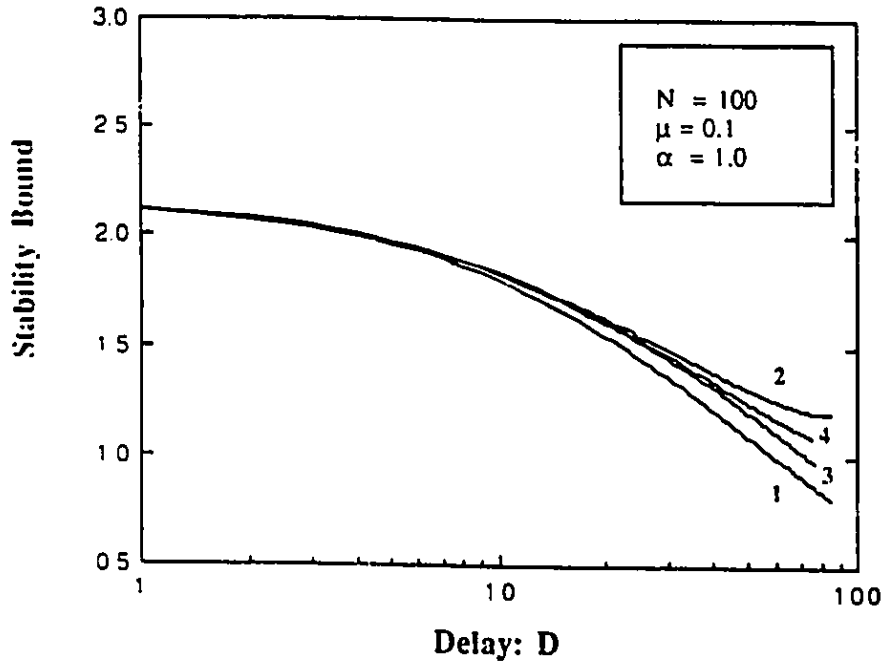


Figure 3.16: Stability bounds ($N\beta\sigma_x^2$) vs Delay with $\mu = 0.1$ and $N = 100$: Accuracy approximation of the LDLMS stability bounds on the step size.

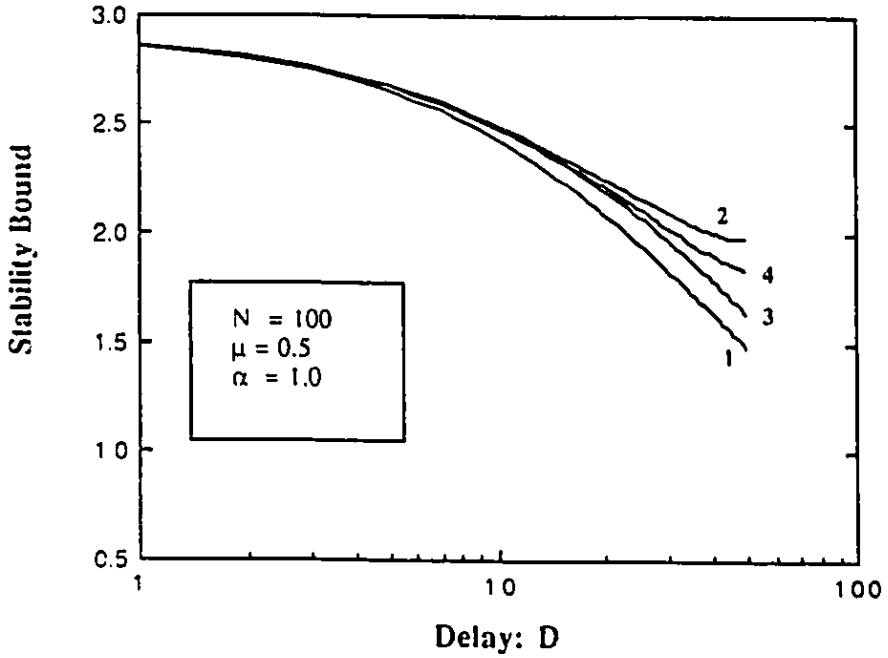


Figure 3.17: Stability bounds ($N\beta\sigma_x^2$) vs Delay with $\mu = 0.5$ and $N = 100$: Accuracy approximation of the LDLMS stability bounds on the step size.

3.6 Stability Bounds of LMS, LLMS, and DLMS Algorithms

The derived expression $J_{ex}(\infty)$ is general in the sense that steady state excess MSE of other well known algorithms can be obtained as special cases of (3.59). In the following sections, it will be shown that by proper substitution of leakage and delay in equation (3.59), expressions for the steady state excess MSE for LMS, LLMS, DLMS are obtained. The performance of all those algorithms is compared with that of the LDLMS.

3.6.1 LMS Algorithm

Let both $\mu = 0$ and $D = 0$, (3.59) can be reduced to the steady state excess MSE of the LMS algorithm and is given by:

$$J_{ex}(\infty) \cong \frac{\beta J_{min}\alpha N\sigma_x^2}{2 - \beta\alpha\sigma_x^2(N + \nu_x - 1)} \tag{3.68}$$

The upper bound on the step size is given directly from the denominator of the previous equation as

$$0 < \beta < \frac{2\sigma_x^2}{\alpha\sigma_x^2(N + \nu_x - 1)}. \tag{3.69}$$

or

$$0 < \beta < \frac{2\sigma_x^2}{\alpha\sigma_x^2 M}. \tag{3.70}$$

These two expressions are identical to the ones derived by W. Gardner in [27]. This maximum permissible step-size for the estimated gradient LMS algorithm is reduced by a factor of the order of M, which is a function of the number of tap-weights N and the kurtosis ν_x of the input signal. It is worth showing the effects of ν_x on the steady state excess MSE and the upper bound of the step size β_{max}. From Figure 3.21, we note that as ν_x increases the upper bound of the LMS algorithm decreases. Hence, the steady state excess MSE decrease with the increase of ν_x. This decrease of $J_{ex}(\infty)$ and β_{max} of the LMS is not noticeable for a larger filter order.

If we compare this bound with the following one,

$$0 < \beta < \frac{2}{\lambda_{max}}, \tag{3.71}$$

We see that the bound in (3.69) is much tighter than (3.71). Also, it is more practical than (3.71) since in most cases λ_{max} is not known and if it was, it would be very costly to compute. The convergence of the MSE based on (3.69) automatically ensures the convergence of the tap-weight filters.

3.6.2 LLMS algorithm

Here the steady state excess MSE of the LLMS algorithm is derived in the general case as well as the the stability bounds on the step size as a function of the leakage parameter. Then, we compare the derived expressions with those of LDLMS algorithm.

To derive the steady state excess MSE of the LLMS, we utilize a similar approach as in the case of the LDLMS. Thus, $J_{ex}(n)$ is given as,

$$J_{ex}(n) = < \epsilon(n)^T H \epsilon(n) > \tag{3.72}$$

where the tap-error is denoted by

$$\epsilon_n = C_n - C_{opt} \tag{3.73}$$

By using the transformation techniques, equations (3.3) and (3.72) can be rewritten as:

$$W_n = (1 - \beta\mu)W_{n-1} + \beta e_{opt}(n)U_n - \beta U_n U_n^T W_{n-1} - \beta\mu R_o \tag{3.74}$$

$$J_{ex}(n) = < W_n^T \Lambda W_n > \tag{3.75}$$

where $e_{opt}(n)$ is the instantaneous error when the tap-weights reach their optimum settings. Now, $J_{ex}(n)$ can be expressed as

$$\begin{aligned}
J_{ex}(n) &= < W_n^T \Lambda W_n > \\
&= (1 - \beta\mu)^2 < W_{n-1}^T \Lambda W_{n-1} > + \beta^2 < e_{opt}^2(n) U_n^T \Lambda U_n > \\
&\quad + \beta^2 < W_{n-1} U_n U_n^T \Lambda U_n U_n^T W_{n-1} > \\
&\quad + \beta^2\mu^2 R_o^T \Lambda R_o - (1 - \beta\mu)\beta < W_{n-1}^T U_n U_n^T \Lambda W_{n-1} > \\
&\quad + \beta^2\mu < W_{n-1}^T U_n U_n^T \Lambda R_o > - \beta\mu(1 - \beta\mu) < R_o^T \Lambda W_{n-1} > \\
&\quad + \beta(1 - \beta\mu) < e_{opt}(n) U^T \Lambda W_{n-1} > - \beta^2\mu < e_{opt}(n) U_n^T \Lambda R_o > \\
&\quad - \beta^2 < e_{opt}(n) U_n^T \Lambda U_n U_n^T W_{n-1} > \\
&\quad + \beta^2\mu^2 R_o^T \Lambda R_o - (1 - \beta\mu)\beta < W_{n-1}^T U_n U_n^T \Lambda W_{n-1} > \\
&\quad - \beta\mu(1 - \beta\mu) < W_{n-1}^T \Lambda R_o^T > \\
&\quad + \beta(1 - \beta\mu) < W_{n-1}^T \Lambda e_{opt}(n) U_n > - \beta^2\mu < R_o^T \Lambda e_{opt}(n) U_n > \\
&\quad - \beta^2 < W_{n-1}^T U_n U_n^T \Lambda e_{opt}(n) U_n >
\end{aligned} \tag{3.76}$$

1. The first term of equation (3.76) is obviously equal to

$$< W_{n-1}^T \Lambda W_{n-1} > = J_{ex}(n-1). \tag{3.77}$$

2. The 2^{nd} term in equation (3.76) can be reduced tc the following expression:

$$\begin{aligned} &< e^2_{opt}(n) U_n^T \Lambda U_n >=< e^2_{opt}(n) >< \sum_{i=0}^{N-1} \lambda_i u_i^2(n) > \\ &= J_{min} N \lambda^2_{rms} = J_{min} \alpha N \sigma_x^4 \end{aligned} \tag{3.78}$$

where λ_{rms} is already defined.

3. The 3^{rd} term becomes:

$$< U_n U_n^T \Lambda U_n U_n^T >=< U_n \sum_{i=1}^{N} \lambda_i u_i^2(n) U_n^T > . \tag{3.79}$$

If we consider only a single element of this matrix, which is given by

$$\begin{aligned} &< \sum_{i=1}^{N} u_k(n) u_m(n) \lambda_i u_i^2(n) > \\ &= \delta_{km} \left[< \sum_{i=1}^{N} u_k^2(n) >< u_i^2(n) > + (\nu_x - 1) \lambda_k < u_k^2(n) >^2 \right] \\ &= \delta_{km} \left[(\sum_{i=1}^{N} \lambda_i^2) \lambda_k + (\nu_x - 1) \lambda_k^3 \right] \end{aligned} \tag{3.80}$$

Therefore, the 3^{rd} term can be eventually reduced to

$$\begin{aligned} &< W_{n-1}^T U_n U_n^T \Lambda U_n U_n^T W_{n-1} > \\ &= N \lambda^2_{rms} < W_{n-1}^T \Lambda W_{n-1} > + (\nu_x - 1) < W_{n-1}^T \Lambda^3 W_{n-1} > \end{aligned} \tag{3.81}$$

The first term of equation (3.81) is equal to $N \lambda^2_{rms} J_{ex}(n-1)$. In addition, an approximation is applied to the second term as follows:

$$\begin{aligned} &(\nu_x - 1) < W_{n-1}^T \Lambda^3 W_{n-1} > \\ &= (\nu_x - 1) < W_{n-1}^T \Lambda^2 \Lambda W^T(n-1) > \\ &\approx \lambda^2_{rms} (\nu_x - 1) < W_{n-1}^T \Lambda W^T(n-1) > \\ &= (\nu_x - 1) \lambda^2_{rms} J_{ex}(n-1). \end{aligned} \tag{3.82}$$

Thus, summing all these terms, we find

$$\begin{aligned} &< W_{n-1}^T \Lambda^3 W_{n-1} > \\ &= N \lambda^2_{rms} J_{ex}(n-1) + \lambda^2_{rms} (\nu - 1) J_{ex}(n-1) \\ &= \alpha M \sigma_x^4 J_{ex}(n-1) \end{aligned} \tag{3.83}$$

where α, λ_{avg}, and ν_x are previously defined.

4. The fourth term of equation (3.76) is a constant and it can be simplified to

$$R_o^T \Lambda R_o = P^T H^{-1} P \tag{3.84}$$

5. The fifth term of the main equation is also reduced as

$$\begin{aligned} < W_{n-1}^T U_n U_n^T \Lambda W_{n-1} > &= < W_{n-1}^T \Lambda^2 W_{n-1} > \\ &= \sum_{i=0}^{N-1} \lambda_i^2 < w_i(n-1) w_i(n-1) > . \end{aligned} \tag{3.85}$$

A approximation on this bound is made and it is given by

$$\begin{aligned} < W_{n-1}^T U_n U_n^T \Lambda W_{n-1} > = &< W_{n-1}^T \Lambda^2 W_{n-1} > \\ \approx &\lambda_{avg} < W_{n-1}^T \Lambda W_{n-1} > \end{aligned} \tag{3.86}$$

6. The 8^{th}, 9^{th}, and 10^{th} terms are zero since $< e_{opt}(n) >= 0$.

7. Finally, identical terms of the main equation are simplified in the same way as before. The overall equation is now written by grouping all the simplified terms,

$$\begin{aligned} J_{ex}(n) \approx\ & (1-\beta\mu)^2 J_{ex}(n-1) + \beta^2 \alpha J_{min} N \sigma_x^4 + \alpha M \sigma_x^4 \beta^2 J_{ex}(n-1) \\ & + \mu^2 \sigma^2 P^T H^{-1} P - 2\beta\sigma_x^2 J_{ex}(n-1) + 2\beta^2 \mu \sigma_x^4 < R_o^T W_{n-1} > \\ & - 2\beta\mu(1-\beta\mu)\sigma_x^2 < R_o^T W_{n-1} > \end{aligned} \tag{3.87}$$

We consider the steady state excess MSE when the algorithm converges, and assuming at the steady state

$$J_{ex}(n) \approx J_{ex}(n-1) \approx \cdots \approx J_{ex}(\infty), \tag{3.88}$$

we obtain the following result of the steady state excess MSE:

$$J_{ex}(\infty) \cong \frac{\beta J_{min} \alpha N \sigma_x^4 + \beta \mu^2 P^t H^{-1} P + 2\beta\mu\sigma_x^2 P^T \epsilon}{2(\sigma_x^2 + \mu) - \beta(\sigma_x^2 \mu + \alpha \sigma_x^4 M + \mu^2)} \tag{3.89}$$

and the stability bound on the step size is given by

$$0 < \beta < \frac{2(\sigma_x^2 + \mu)}{\sigma_x^2 \mu + \alpha \sigma_x^4 M + \mu^2} \tag{3.90}$$

This is identical to the equation derived for the LDLMS algorithm for zero delay (3.59).

The effect of ν_x on the LLMS stability bound and on the steady state excess MSE are the same as for the ones of the LMS algorithm. However, for the case of the LLMS, an increase in the residual error results, i.e.,

$$J_{ex_{(LLMS)}}(\infty) = J_{ex_{(LMS)}}(\infty) + (C_{opt_\mu} - C_{opt})^T R(C_{opt_\mu} - C_{opt}) \tag{3.91}$$

where C_{opt_μ} is the optimum tap-weight vector of the LLMS as a function of μ. The tolerable range of the final residual error of the LLMS depends upon applications.

3.6.3 DLMS Algorithm

The steady state excess MSE is deduced from (3.59) by putting μ equal zero. So, we obtain,

$$J_{ex}(\infty) \cong \frac{J_{min}\alpha N\beta\sigma_x^2}{2 - (\alpha M + 2D)\sigma_x^2\beta + D(D+1)\sigma_x^4\beta^2 - \frac{1}{3}D(D+1)(2D+1)\sigma_x^6\beta^3 + \ldots} \tag{3.92}$$

The stability bound for the DLMS algorithm is a function of the delay and is solved numerically. The excess MSE obtained here, as a special case of LDLMS with zero leakage, is exactly the same as in [4]. The bounds on the step size are then found using numerical methods. From this equation, we can see that only a small increase in the steady state excess error results, when $\frac{D}{\alpha M} \ll 1$. The coefficient of the higher powers of β in (3.57) (. > 4) for zero leakage do not have closed form expressions as in the case of powers of lower than 4 and would require that the summations be computed explicitly. These high power terms are to be unnecessary because the contributions from the lower power terms in expression (3.57) are dominant. In our derivation for the LDLMS algorithm, a 4^{th} power order term of β is derived and included in the expression (3.59) while in equation (3.92) [4] does not have this term. So, our derivation is more complete since it includes an additional term.

It is worth examining the effects of delay on the the steady state excess MSE. Figure 3.18 shows a slight increase in the excess error when the delay increases. We note

also that $J_{ex}(\infty)$ is slightly smaller for a Gaussian distribution than for binary symmetrical distribution. This shows that adaptive algorithms perform differently under distinct input probability density functions.

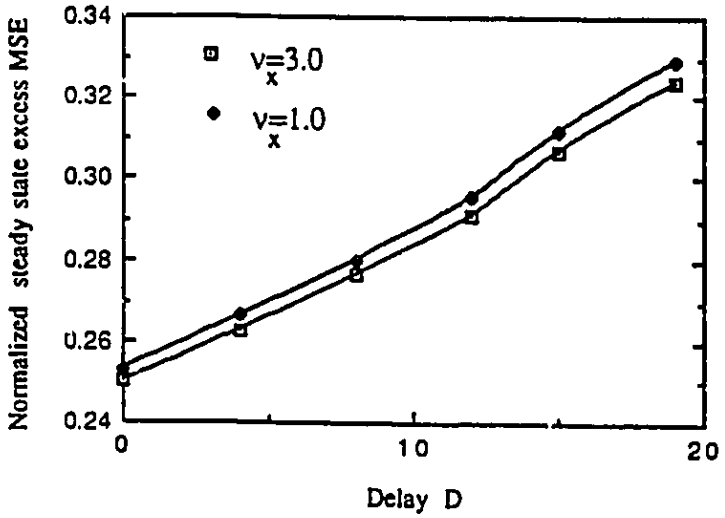


Figure 3.18: Effects of delay on the steady state excess MSE of the DLMS algorithm. the steady state excess MSE is normalized to $\frac{J_{ex}(\infty)}{J_{min}}$. $N = 40$, $\alpha = 1$, $\beta = 0.01$ and $\sigma_x^2 = 1$.

3.7 Comparison of the MSE Convergence Behavior of LMS, LLMS, DLMS, and LDLMS Algorithms

The excess MSE of the LMS algorithm can be given from equation (3.88) for zero leakage ($\mu = 0$) as:

$$J_{ex}(n) = (1 - 2b + \alpha b^2 M) J_{ex}(n-1) + J_{min} \alpha N b^2, \tag{3.93}$$

where $b = \sigma_x^2 \beta$ and $M = N + \nu_x - 1$. Considering only the homogeneous part of (3.93), the characteristic equation can be given by

$$J_{ex}(z) = (1 - 2b + b^2 M) z^{-1} J_{ex}(z)$$

or

$$1 - (1 - 2b + b^2 M)z^{-1} = 0, \tag{3.94}$$

where z is the Z-transform. The stability and convergence of the algorithm can be explained by considering the roots of (3.94). There is only one real root given by

$$z = 1 - 2b + b^2 M. \tag{3.95}$$

The variation of the root magnitude of (3.95) versus the step size b determines the MSE convergence speed. When the step size b is within its upper bound, the system is stable. Also, the convergence of the MSE does not exhibit any oscillations and it decays monotically. Figure 3.19 shows root magnitude of (3.95) versus step size b for the case of the LMS algorithm. From that Figure, the root magnitude has a minimum in the middle of the interval. This indicates that if b equals half its upper bound, the MSE has the fastest speed of convergence.

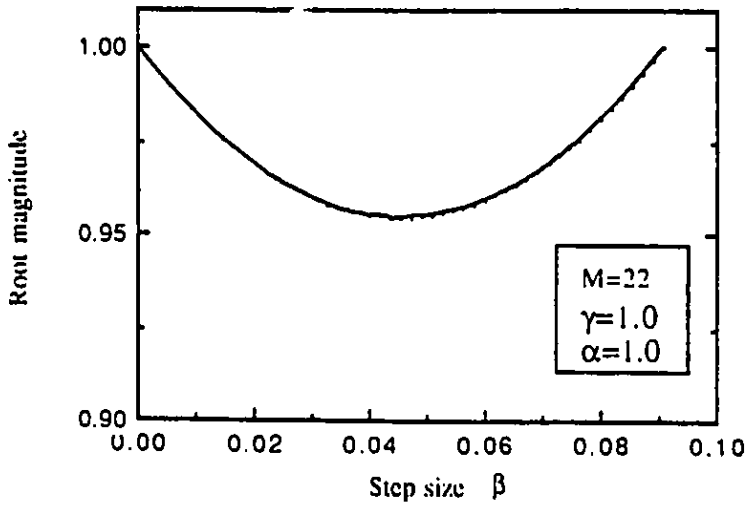


Figure 3.19: Root magnitude vs step size for the LMS characteristic equation: M=22.

In the case of the DLMS, the characteristic equation can be derived to study the MSE convergence speed. We can assume that for $\frac{D}{N} \ll 1$, equation (3.57) for zero leakage can be reduced to

$$J_{ex}(n) = J_{ex}(n-1) + \beta^2 \alpha J_{min} N \sigma_x^4 + \alpha M \sigma_x^4 \beta^2 J_{ex}(n-D-1)$$

$$-2\beta\sigma_x^2 J_{ex}(n-D-1) + 2\beta^2\sigma_x^4 \sum_{j=1}^{D} J_{ex}(n-D-j-1) \tag{3.96}$$

The characteristic equation (3.96) is given by

$$\begin{aligned} & 1 - \left[z^{-1} + \alpha M b^2 z^{-D-1} - 2bz^{-D-1} + 2b^2 \sum_{j=1}^{D} z^{-D-j-1} \right] \\ = \quad & 1 - \left[z^{-1} + b(\alpha M b - 2 + 2b\frac{z^{-1} - z^{-D-1}}{1 - z^{-1}}) z^{-D-1} \right] \\ = \quad & z^{D+1} - 2z^D + z^{D-1} + 2b^2 z^{-D-1} - 2b^2 z^{-1} - b(\alpha M b - 2) = 0 \end{aligned} \tag{3.97}$$

As we can see, this equation has real roots and complex roots. For convergence of the algorithm, we find a dominant positive real root of (3.97) which is larger than the other complex roots. Hence, any oscillations in the DLMS MSE convergence are small. In addition, The DLMS MSE convergence does not decay monotonically as in the case of the LMS algorithm because of the presence of complex roots. As D increases, the complex roots start to be very important and the convergence behavior of the DLMS MSE becomes oscillatory. Eventually, this will lead to a degradation of performance.

In the case of the LLMS and the LDLMS, the system equations (3.57) and (3.87) have more constant terms than in the case of LMS and DLMS. This comes from the fact that leakage introduces an excess amount of error in the final steady state MSE as it was previously discussed. The above discussions about LMS and DLMS algorithm can be as well applied to LLMS and LDLMS respectively.

The effects of leakage in LMS is to speed-up the MSE convergence. This can be described by the homogeneous part of the approximated system equation (3.87) for small values of leakage:

$$J_{ex}(n) \approx \gamma^2 J_{ex}(n-1) + \alpha M b^2 J_{ex}(n-1) - 2\gamma b J_{ex}(n-1) \tag{3.98}$$

The characteristic equation of (3.98) is given by

$$1 - (\gamma^2 - 2b\gamma + \alpha b^2 M) z^{-1} = 0, \tag{3.99}$$

where for simplicity we denoted the leakage by $\gamma = (1 - \beta\mu)$. The variation of the root magnitude of (3.99) versus b is depicted in Figure 3.20. We see the root magnitude of (3.99) has a minimum value which is smaller than in the LMS case. This means that the LLMS MSE converges much faster the LMS MSE depending on the amount of leakage γ.

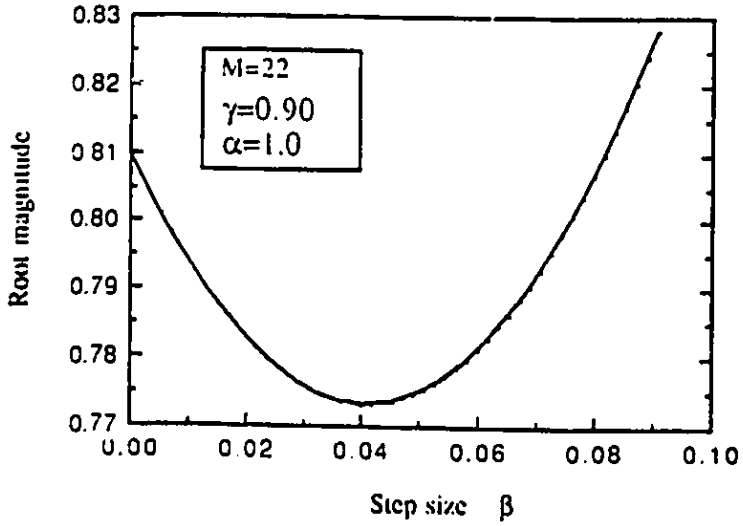


Figure 3.20: Root magnitude vs step size for the LLMS characteristic equation: $M = 22$, $\gamma = 0.90$.

The same conclusions above can be stated for the DLMS and LDLMS algorithms. In the LDLMS algorithm, leakage speeds up the MSE convergence. Because of the existence of complex roots in the DLMS characteristic equation, leakage reduces the oscillatory behavior of the DLMS MSE. In the other hand, the large amount of leakage in the algorithm will lead to performance degradation of the MSE.

3.8 Optimum Step Size for Maximum Rate of Convergence of LDLMS

In many situations, a fast convergence of the error is desirable. In this case, a step size that speeds up convergence has to be found. For system equations with lower step size magnitude, the optimum value that results in fast convergence speed can be easily found. However, because the system equation has a complicated form, an optimization procedure has to be used to find the optimum step size β_{opt}.

Because of this mathematical difficulty, the system equation (3.59) is considered with only second order degree of step size. This in fact offers a flexibility for choosing an appropriate optimum step size with appropriate leakage and delay values.
Let's consider the system equation (3.57). For values of $D\beta \ll 1$, all the approximations in Figures 3.10- 3.17 are the same, so the system equation can be approximated as:

$$\begin{aligned} J_{ex}(n) &= (1-\beta\mu)^2 J_{ex}(n-1) + \beta^2\alpha J_{min} N\sigma_x^4 + \alpha M\sigma_x^4\beta^2 J_{ex}(n-D-1) + \\ &\quad \beta^2\mu^2 P^T H^{-1} P + 2\beta^2\mu\sigma_x^4 < R_o^T W_{n-D-1} > -2\beta\mu(1-\beta\mu)\sigma_x^2 < R_o^T W_{n-1} > \\ &\quad -2\beta(1-\beta\mu)\sigma_x^2 J_{ex}(n-D-1) + 2\beta^2(1-\beta\mu)\sigma_x^4 \sum_{j=1}^{D} J_{ex}(n-D-j-1) \end{aligned} \tag{3.100}$$

Before finding the optimum step size, the following assumptions are employed to simplify the analysis:

- The algorithm is far from the optimum. The quantities J_{min} and $J_{ex}(n)$ are generally unknown. However, during early stages of MSE convergence, the following assumption can be used,

$$J_{ex}(n), J_{ex}(n-1), J_{ex}(n-D-1), \ldots, \gg J_{min}.$$

- When the step size is small enough, we can assume during early stages of MSE convergence that the excess MSEs are assumed to be approximately equal,

$$J_{ex}(n) \approx J_{ex}(n-1) \approx J_{ex}(n-D-1) \approx \ldots \approx J_{ex}(n-2D-1)$$

- The fourth, fifth, and sixth terms are assumed to be small compared to other terms since the step size and leakage values are small.

Differentiating (3.100) with respect to β, we get :

$$\begin{aligned}\frac{\partial J_{ex}(n)}{\partial \beta} &= 2\mu(1-\beta\mu)(-2\beta\mu)J_{ex}(n-1) + 2\alpha M\sigma_x^4\beta J_{ex}(n-D\cdot\cdot 1) \\ &-2(1-\beta\mu)\sigma_x^2 J_{ex}(n-D-1) + 2\beta\mu\sigma_x^2 J_{ex}(n-D-1) \\ &+4\beta^2\sigma_x^4(1-\beta\mu)\sum_{i=1}^{D} J_{ex}(n-D-j-1) - 2\beta^2\mu\sigma_x^4\sum_{j=1}^{D} J_{ex}(n-D-j-1)\end{aligned} \tag{3.101}$$

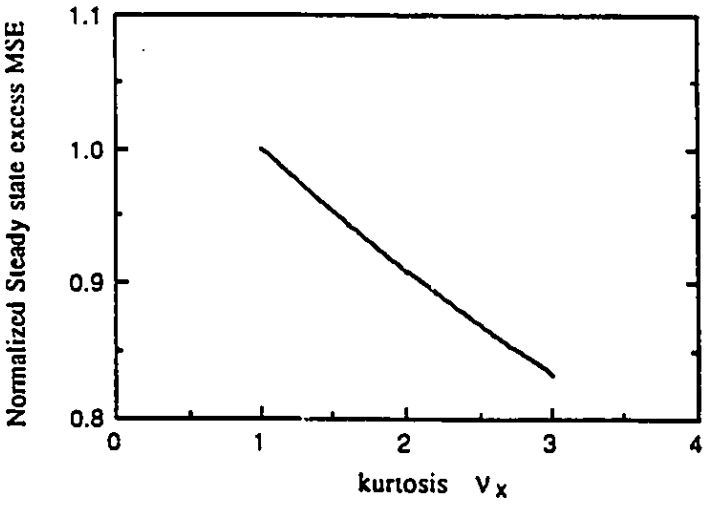
After a few simplifications, (3.101) becomes

$$\mu^2(1-\sigma_x^4 D)\beta_{opt}^2 + (\alpha M\sigma_x^4 - 2\mu + 2\mu\sigma_x^2 + 2\sigma_x^4 D)\beta_{opt} - \sigma_x^2 = 0. \tag{3.102}$$

This second order equation can be solved. The optimum step size versus leakage parameter μ is plotted in Figure 3.22- 3.23 for different kurtosis values ν_x. It can be seen that for inputs with large kurtosis $\nu_x = 3$, β_{opt} is smaller than in the case of inputs $\nu_x = 1$. Also, we see that the optimum step size changes slightly with leakage for both cases.

3.9 Theoretical and Experimental Bounds of LDLMS

The analytical results of the stability bounds of the LDLMS algorithm presented here have been verified by computer simulations, where an adaptive transversal echo canceller has been implemented. The echo signal is generated as the output of a 6^{th} order low pass infinite impulse response (IIR) filter with white Gaussian noise as input. The order of the adaptive filter was 40 taps. The echo impulse response of the IIR filter used is depicted in Figure 3.24. Experimental results for the derived stability bounds are obtained by increasing the step size β until the algorithm diverges. Figures 3.26- 3.25 show the theoretical bounds obtained from (3.60) and experimental bounds are quite close. For smaller values of leakage μ, simulation results are fairly close to the analytical ones.



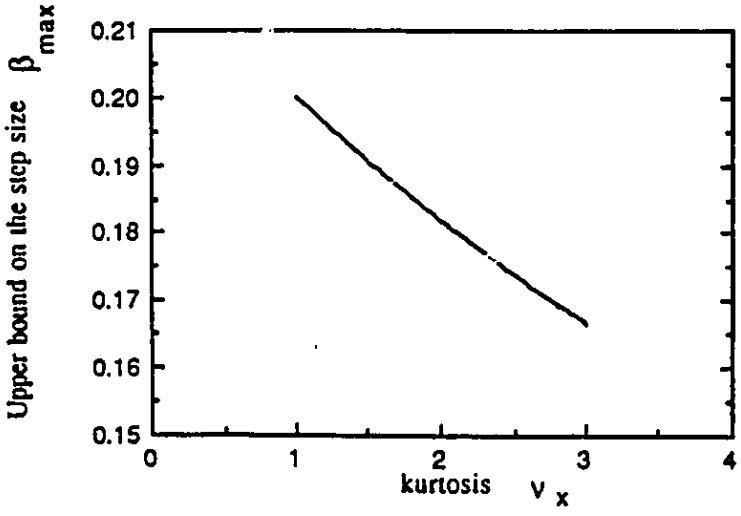
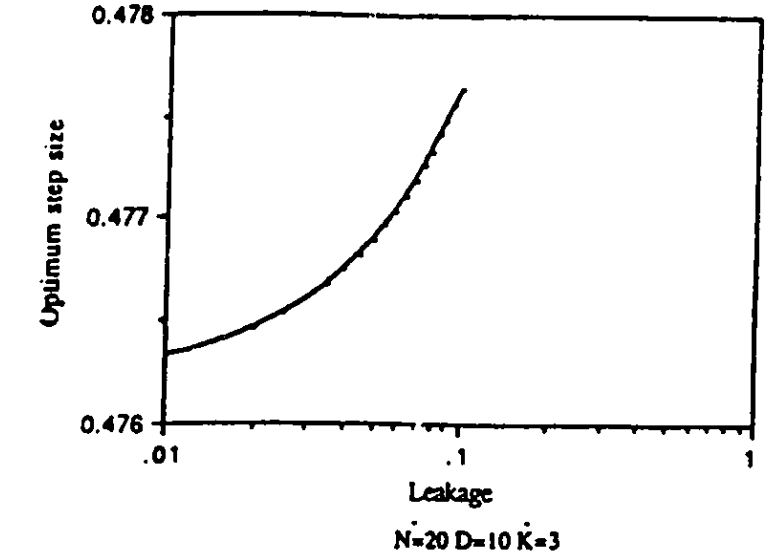


Figure 3.21: Effects of kurtosis ν_x on the steady state excess MSE and the maximum step size of the LMS algorithm. $J_{ex}(\infty)$ is normalized to $\frac{J_{ex}(\infty)}{J_{min}}$. $N = 10$,$\alpha = 1$, and $\sigma_x^2 = 1$.



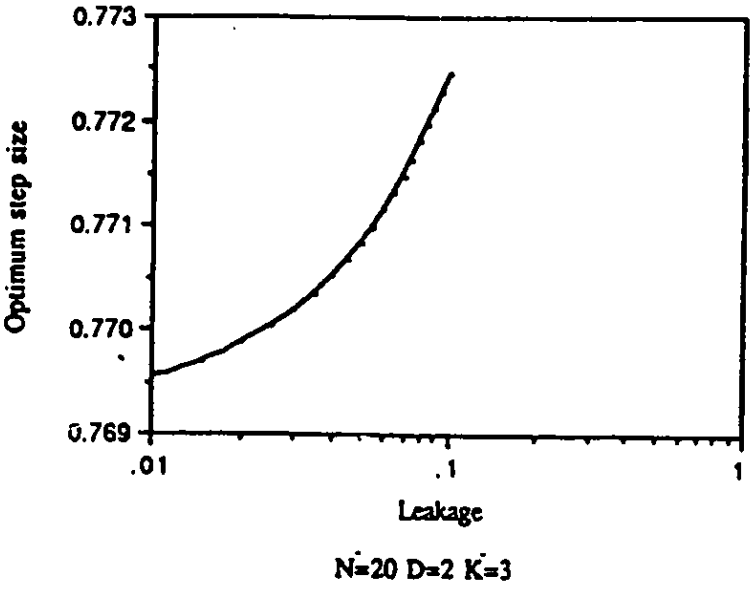
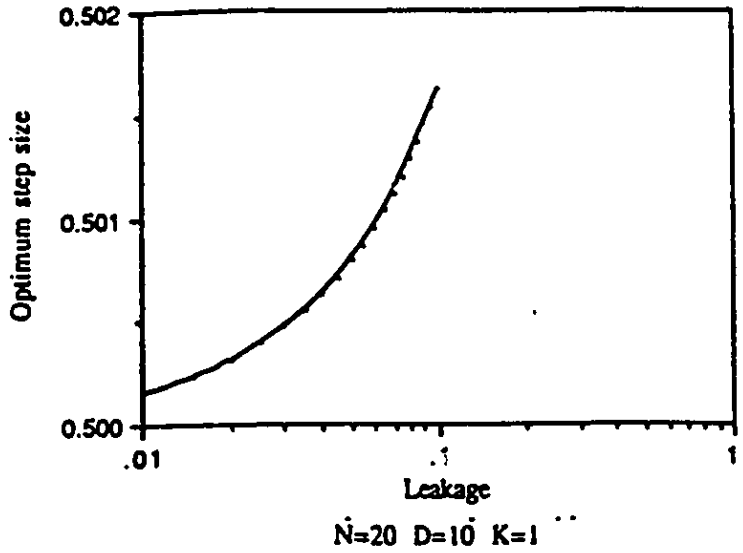


Figure 3.22: Optimum Step size of LDLMS for fastest convergence speed vs leakage μ for different values of delay D and kurtosis ν_x (ν_x=K).



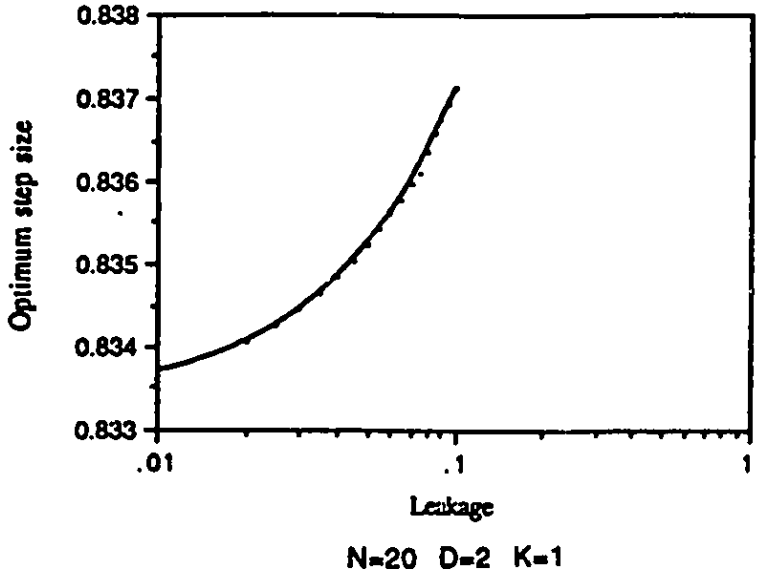


Figure 3.23: Optimum Step size of LDLMS for fastest convergence speed vs leakage μ for different values of delay D and kurtosis ν_x (ν_x=K).

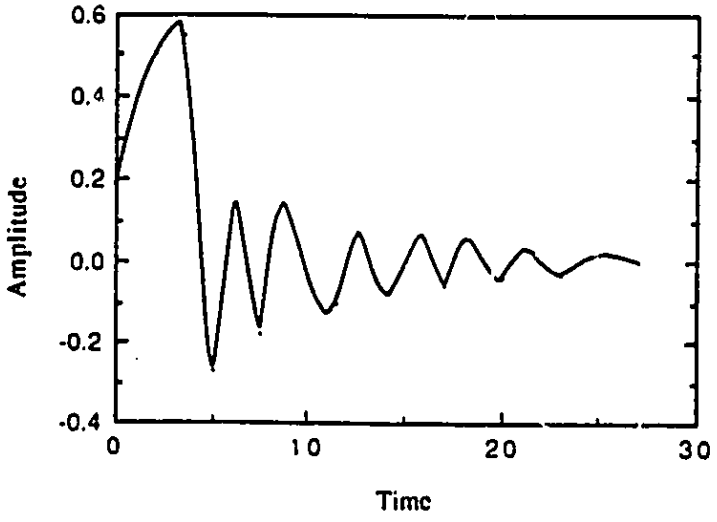


Figure 3.24: Infinite Impulse response(IIR) filter used in simulation for verification of the theoretical stability bounds of the LDLMS algorithm.

3.10 Computational Complexity of the LDLMS

The number of multiplications and additions needed to perform one iteration of the adaptive algorithms is used as a measure of computational complexity. In practical implementation of the adaptive algorithms, the step size is chosen to be a power of two so that the update of each term of $C(n)$ can be done by shifting the correction term to accomplish the β multiplication. In this case, the LMS, DLMS, LLMS, and LDLMS require N real multiplications and $N-1$ additions to update $C(n)$ for each iteration, where N is the length of the filter impulse response. On the other hand, the values of leakage parameter like the step size are generally restricted to power of two in order to prevent a full multiplication. The factor $(1-\beta\mu)$ in either LLMS or LDLMS may lead to the introduction of a round-off noise, thus an increase of one bit in the coefficient wordlength may result [14].

3.11 Discussion

The convergence and stability bounds of the LDLMS algorithm are examined. For the stability bound based on the convergence of the filter coefficients, it was shown how leakage prevents tap-weight drift by ensuring that the poles of the system equation stay within the unit circle. The introduction of the appropriate leakage can be used to guarantee the stability of the algorithm. However, the forced response term in the system equation of the LDLMS algorithm imposes a constraint on the required performance. A bias in the optimum weight setting is added. This amount of bias can be controlled by a careful design of leakage and it depends on the particular application.

The stability bound of the LDLMS based on the error convergence has also been derived. It has been shown that the residual output error power expression has a complicated form because of the presence of the leakage parameter. However, for smaller values of leakage, the term including the leakage may be neglected since in many applications [5], [8],and [14], small values of leakage are chosen and proven to be very effective for a required performance.

The convergence properties of$< e^2(n) >$ and $< W(n) >$were seen to be different. The convergence speed of $< e(n)^2 >$ depends mainly on the number of taps while that of $< W_n >$ does not. For most practical cases, the filter length will be the dominating factor. Thus, there exists values of step size, β, for which $< W_n >$ converges while $< e(n)^2 >$ diverges. Thus, if both quantities converge, $< W_n >$ will generally converge faster than $< e(n)^2 >$. Hence, the stability bound of $< e(n)^2 >$ implies a much tighter upper bound on the step size than the one on $< W(n) >$.

The stability bound on the step size is a function of delay, leakage, and other terms. An optimized value of leakage for a specific application can be found directly either by using the Figures 3.10- 3.17 or solving the polynomial $S(\beta)$ numerically. Usually, the ratio $\frac{D}{N} \ll 1$, so only up to the 4^{th} order terms of $S(\beta)$ are sufficient since other high order terms do not contribute considerably.

The stability bound is derived based on the convergence of $< e^2(n) >$, which is

tighter than the one based on coefficient convergence. The results so far derived and discussed do not have to be restricted only to system identification. They can be easily extended to various applications such as decision feedback equalizers [27].

3.12 Conclusion

An approximation of the stability bound on the step size of the LDLMS algorithm was provided by (3.57). The bounds are in terms of the number of taps, the input power signal, the input kurtosis distribution, the leakage parameter, the eigenvalue spread, and the delay. The stability bound of the adaptive tap-weight has also been investigated by the root locus technique and it has been shown how leakage prevents the poles of the characteristic equation of the LDLMS algorithm from drifting out of the unit circle. In addition, step size that results in fast convergence speed was derived for different values of leakage parameter. It was shown that the orders of the approximation of the stability bounds are fairly good as long as the ratio $\frac{D}{N}$ is very close to one. The steady state excess MSE and the stability bounds derived are general expressions in the sense that expressions of other well known algorithms (LMS, DLMS, LLMS) can be deduced by substitutions of leakage and/or delay =0. The bound on the step size for error convergence has been verified by simulations and shown to be fairly close to the derived one.

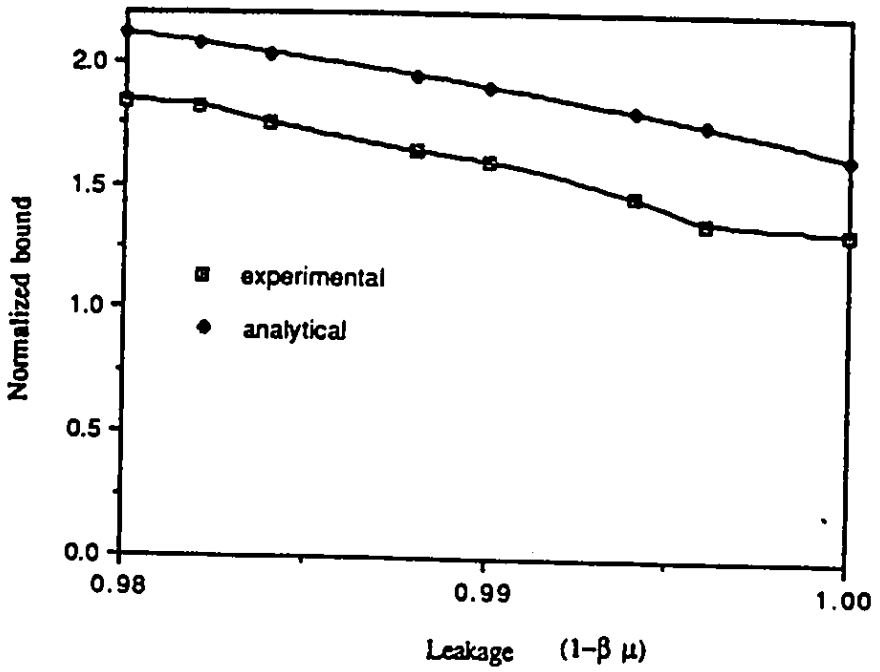
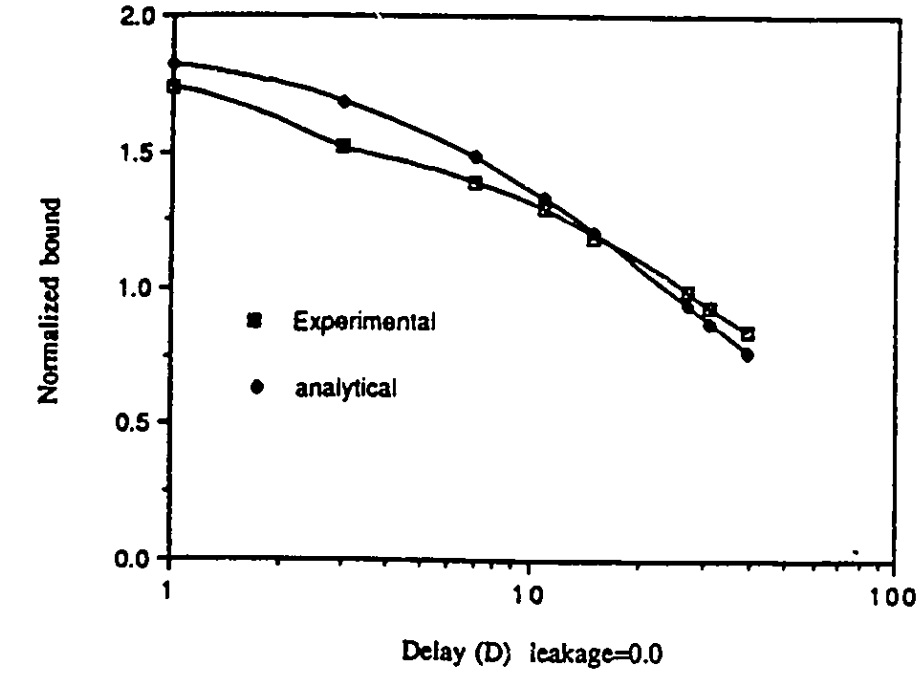


Figure 3.25: Comparison of theoretical and experimental results of the LDLMS stability bound for different values of leakage γ ($\gamma = 1 - \mu\beta$ and $N\beta\sigma_x^2$ is termed as normalized bound) D=4, N=40



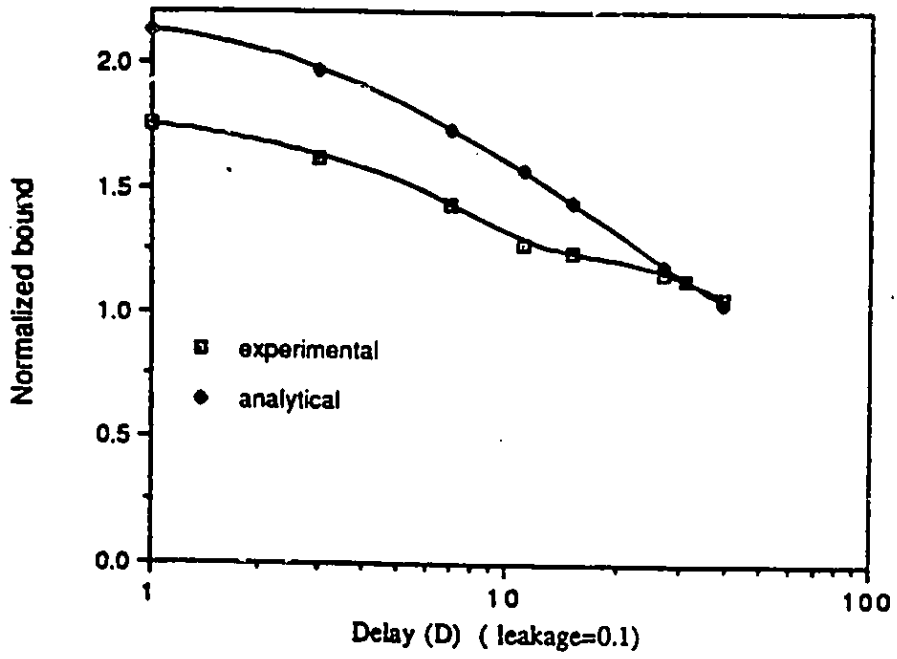


Figure 3.26: Comparison of theoretical and experimental results of the LDLMS stability bound for different values of delay D ($N\beta\sigma_x^2$ is termed as normalized bound) : $N = 40$

Chapter 4

Applications of the Leaky Delayed LMS Algorithm

The LDLMS algorithm is used in a large variety of applications. In this chapter, we consider only a system identification model. The task here is to identify an unknown system by adaptively adjusting the tap-weights $C(n)$. These tap-weights should eventually converge to the unknown system parameters.

Three examples of the LDLMS algorithm applications are given. In the first example, a signal with little energy content over part of its frequency band is used as an input. Because of such an input, the parameter estimates of the adaptive filter, when using the LMS or the DLMS algorithms, drift away from the desired values despite the bounded inputs and bounded estimation errors. On the other hand, employing the LLMS algorithm, the tap weights will be bounded and the drifting phenomena has been shown to disappear. This drifting phenomena is well explained in [13] and [22]. The drifting type that is devoted in this chapter is due to the lack of the persistent excitation on the input process. The persistent excitations are obeyed if the mean of the input process is finite and the input correlation H is positive definite. In other words, the frequency spectrum of the input process has to cover all the frequency rang of the unknown channel to be identified.

The performance of the LLMS in controlling the drifting phenomena is examined

here for the more realistic situation including the effect of the delay.

In the second example, we consider the situation when the input signal to the adaptive filter or the gradient estimates of the gradient adaptive algorithm, $e_{k-D}X_{k-D}$, vanishes (i.e. equals zero). Then, coefficients of the adaptive filter utilizing the DLMS algorithm are locked-up. This phenomena is known as stalling or lock-up situation and it is encountered in several applications of adaptive algorithms [5]. This stalling phenomena happens when the DLMS algorithm stops updating. Introducing the LDLMS, the leakage factor reduces the magnitude of the adaptive weights in a recursive fashion, and then the locked-up tap-weights leak to zero. Again, the LDLMS performance is studied to see the effect of leakage on controlling the stalling phenomenon.

The last example is to examine the tracking capability performance of the LMS and DLMS algorithms under a time-varying system. In this nonstationary environment, the effect of delay on the algorithm misadjustment is examined for various step size values with or without leakage. Then, a comparison of the stability and the LMS, LLMS, DLMS, and LDLMS algorithm misadjustments is investigated through simulation. Finally, we end this chapter with discussion and conclusions.

4.1 Adaptive algorithm with Spectral Deficiency Inputs

Parameter drift is encountered in several practical applications of adaptive algorithms [22] and [36]. The drifting considered in this chapter is due to the lack of the persistent excitation in the input process. The persistent excitations are obeyed if the mean of the input process is finite and the input correlation H is positive definite. In other words, the frequency spectrum of the input process has to cover all the frequency range of the unknown channel to be identified.

Simulations are used to illustrate this phenomenon. First, a weak spectral signal is used as input to the system identification block. The DLMS algorithm is used to identify the unknown system parameters. Since the signal is spectrally poor, weights do

not converge. In contrast, utilizing the LDLMS algorithm, all the weights are stabilized and they converge faster depending on the amount of leakage which is introduced. As a penalty for using leakage , biases are introduced on the final values of the tap-weights. In other words, the final optimum tap-weights satisfy

$$\lim_{n\to\infty} C(n) - C_{opt} \neq 0 \tag{4.1}$$

Thus, leakage should be kept as small as possible to avoid large biases while stopping the tap-weight drifting.

The drift phenomena may happen with or without the delay. Small or negligible eigenvalues may cause the drifting problem. To explain the existence of the drifting problem for the LDLMS algorithm, we consider the case of the LLMS algorithm and what we conclude can be identically applied to the case of the LDLMS. The LLMS equation is written here as a function of input correlation matrix H and cross-correlation vector P:

$$C(n) = (\gamma I - \beta H)C(n-1) + \beta P \tag{4.2}$$

where $\gamma = (1-\beta\mu)$. The expression for the tap-weight error vector is derived in Appendix B and is given by

$$\epsilon(n) = \sum_{k=0}^{N-1}(\gamma - \beta\lambda_k)^n V_k^T \epsilon(0) V_k + (\gamma - 1)\sum_{k=0}^{N-1}\frac{1-(\gamma-\beta\lambda_k)^n}{1-(\gamma-\beta\lambda_k)}C_o. \tag{4.3}$$

The last equation shows that the error $\epsilon(n)$ follows a trajectory which is the sum of all the modes of the adaptive algorithm plus an offset term that depends on the leakage factor γ. When $\gamma = 1$, this corresponds to the LMS. Convergence here depends mainly on the step size. In the case of LDLMS, that depends on the bound of the step size already derived. As β increases, $\epsilon(n)$ and the final mean-squared error increase.

On the other hand,$|\gamma - \beta\lambda_k|$ depends on the eigenvalue of the input autocorrelation matrix. The convergence speed of $\epsilon(n)$ is also based on that quantity. In the LMS case, equation (B.7) becomes as

$$\epsilon(n) = \sum_{k=0}^{N-1}(1 - \beta\lambda_k)^n V_k^T \epsilon(0) V_k. \tag{4.4}$$

The value of the right hand term of (4.4) is larger for small or negligible λ_k and the sum of these terms for all the modes of the adaptive filter may get to be very large. Consequently, $\epsilon(n)$ starts to drift far away from zero. With leakage, however, $\epsilon(n)$ can be made smaller so unique optimum values can be reached. Since this term will decay faster depending on the amount of leakage, convergence speed will increase.

There is, however, a tradeoff inherent in using leakage in the persistent excitation. A small bias due to leakage requires γ close to one, but eliminating drifting requires a small γ. A compromise is required since the same leakage is used to every component of the algorithm. An alternative way is to apply leakage only to regions of the input process where drift can occur. However, this way requires more knowledge of the input process than is usually known to the user.

The models used for the simulation are shown in Figure 4.1. Model#1 shows a system identification block with a stationary input process. The input noise is zero mean white Gaussian. However, in Model#2, the zero mean white noise input is first filtered by a low pass filter and then passed through the same system identification as in Model#1. The pre-channel filter is included in system Model#2 (FIR1) and the unknown system parameters (FIR2) used in both models are finite duration impulse responses. The FIR1 is a pre-filter with 12 taps while the FIR2 is of length 22. Their impulse and and frequency responses are shown in Figure 4.2. The adaptive filter has the same order as FIR2. The filter coefficients of FIR1 and FIR2 are given in Appendix C.

Simulation results for Model#1 and Model#2 are presented in the following figures. Figures 4.3- 4.5 show the convergence of the adaptive tap-weights of the DLMS and LDLMS using Model#1 and #2. Here, we have chosen only tap-weight#5,8, 9, 10, 11, and 12 of the adaptive filter for illustrative purposes. Other filter tap-weights behave the same way and similar conclusions can be drawn. From those figures, we can see that the tap-weights in Model#1 converge to the optimum after a few hundreds of iterations. This corresponds to the stable situation of DLMS (Model#1). However, results show when using Model #2, the DLMS filter coefficients drift away from their optimum settings. They exhibit instability as they do not settle down to constant values. This situation is

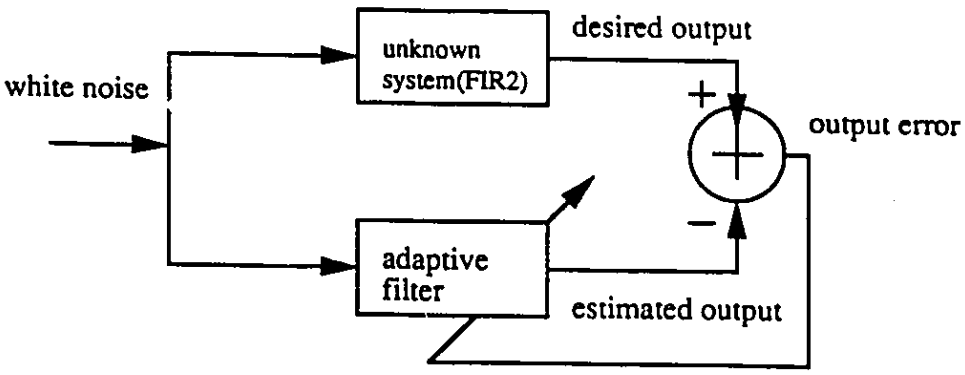
due mainly to poor excitation of the input signal. When we introduce the LDLMS, all the tap-weights converge to steady state settings and do not exhibit any kind of instability as can be shown from the figures. Figures 4.3 through Figures 4.5, small values of leakage are used and they were able to drive the filter coefficients to stability. Convergence and stability of the filter coefficients were guaranteed with $\mu = 0.08$. A smaller value $\mu = 0.04$ can still guarantee the convergence of the LDLMS and it it introduces less bias on the estimated parameters. Therefore, an optimized μ is required to provide compromize between stability and bias.

As it can be seen from the figures, the drifting coefficients do not converge even after a number of iterations orders of magnitude higher than needed for convergence in Model#1. The learning curves of the LDLMS and DLMS are depicted in Figure 4.6 where effects of leakage on the final steady state-excess MSE are shown.

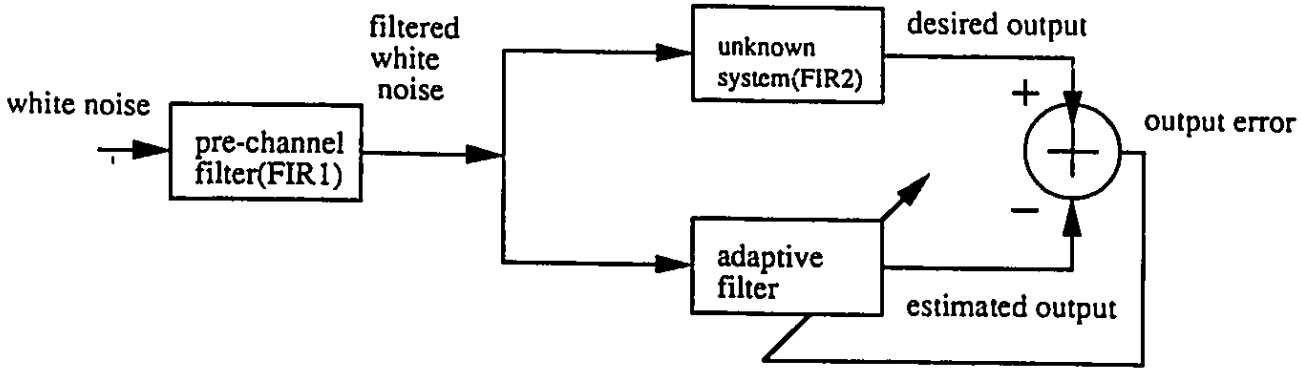
The drifting problem also exists in the regular LMS algorithm. To see that, we applied the LMS and LLMS algorithms to the Model #1 and Model#2. The simulation results for these two algorithms are shown in Figure 4.7 to Figure 4.9 for taps# 1-6. When applying LMS algorithm to Model#1, no stability problems are encountered since this situation corresponds to the ideal input conditions. The instability behavior of the LMS can be noticed when it is applied to Model#2. In contrast, when using the LLMS algorithm, stability can be monitored by the use of a proper leakage value as it is depicted in Figures 4.7 - 4.9.

4.2 Stalling Situation in the Delayed LMS Algorithm

Another type of applications where leakage is used to improve the system performance is when the input signal vanishes or the gradient estimate $[e(k-D)X(k-D)]$ equals zero, the filter coefficients lock-up(i.e.stop adapting). Then, it might be preferable to have the filter weights return to zero [2]. The accumulated quantization noise that results from finite-precision representations of the system parameters may act as an input with



system identification Model#1



system identification Model#2

Figure 4.1: Model#1 and Model #2 used in simulations. Model#1 is for an input with rich frequency content. Model#2 is for an input with low energy content.: FIR1 is the Pre-filter and FIR2 is the unknown system.

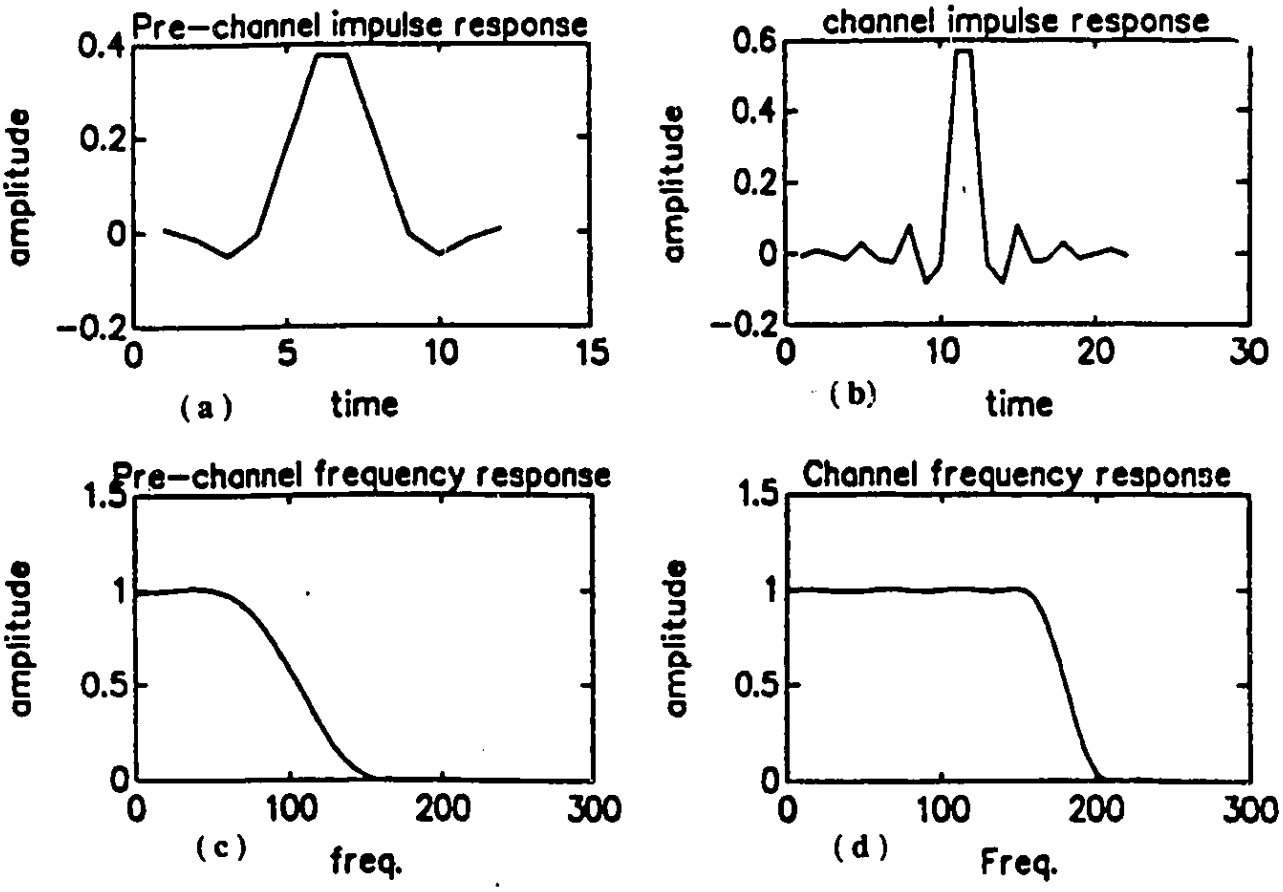


Figure 4.2: Impulse and Frequency Responses of the Pre-filter and Unknown system (a) FIR1 filter $N = 12$ (b) FIR2 filter $N = 22$ (c) frequency response of the pre-channel filter, FIR1 (d) frequency response of the unknown channel response, FIR2.

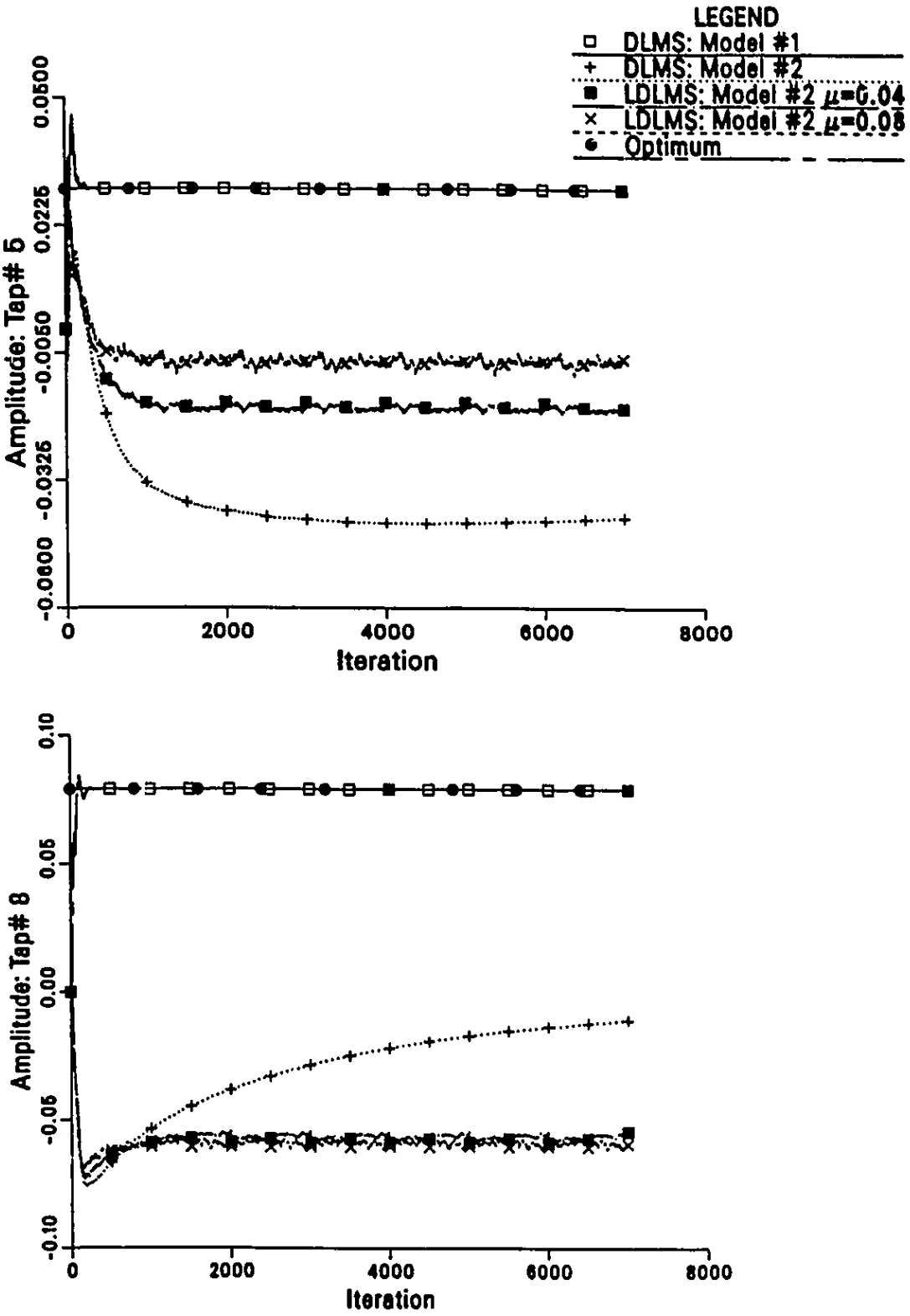


Figure 4.3: The DLMS and LDLMS mean tap-weight convergence using Model#1 and Model#2. Figures show the learning curves of the adaptive taps#5 and #8 for $D = 8$ and $\beta = 0.03$

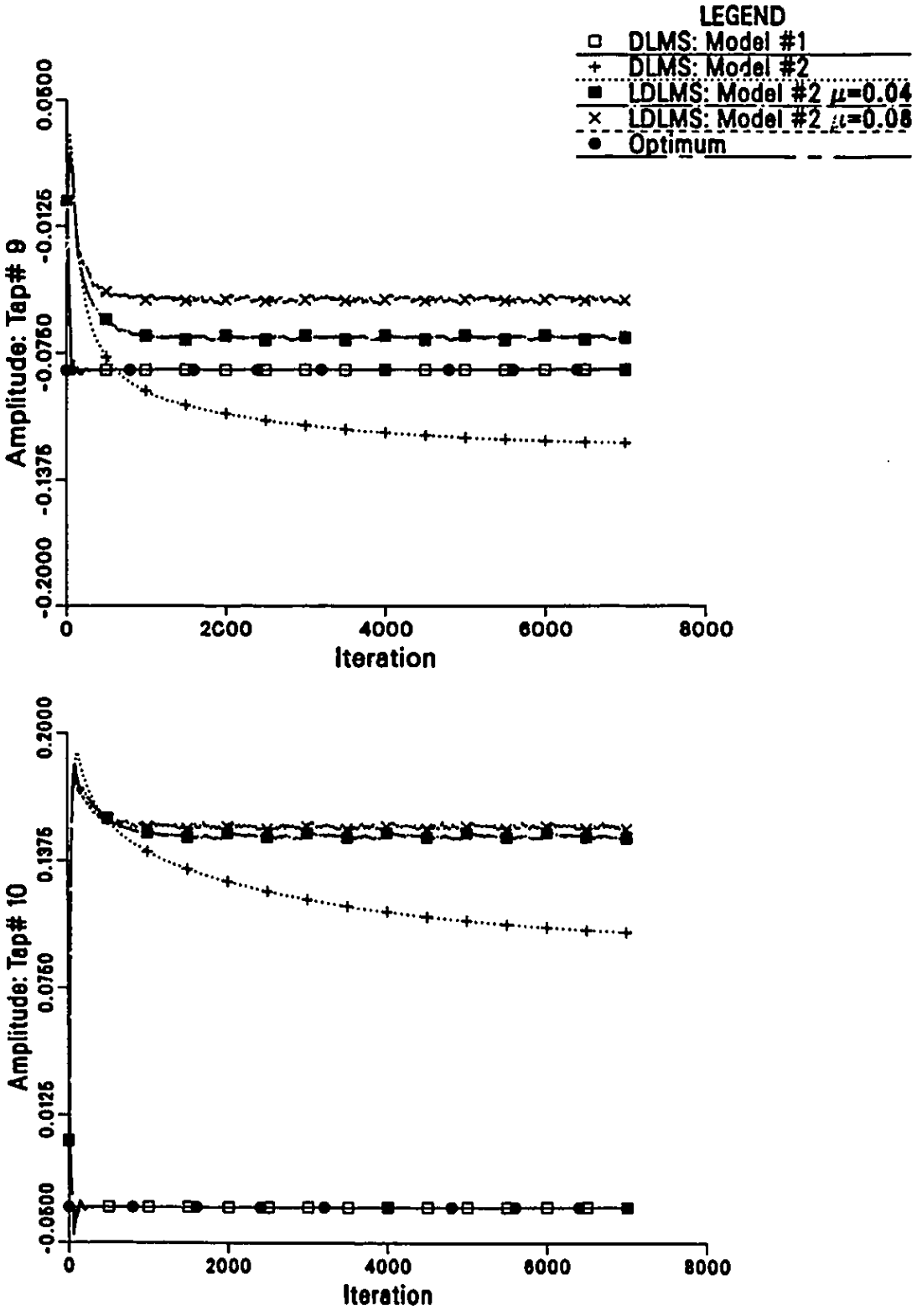


Figure 4.4: The DLMS and LDLMS mean tap-weight convergence using Model#1 and Model#2. Figures show the learning curves of the adaptive taps#9 and # 10 for $D = 8$ and $\beta = 0.03$

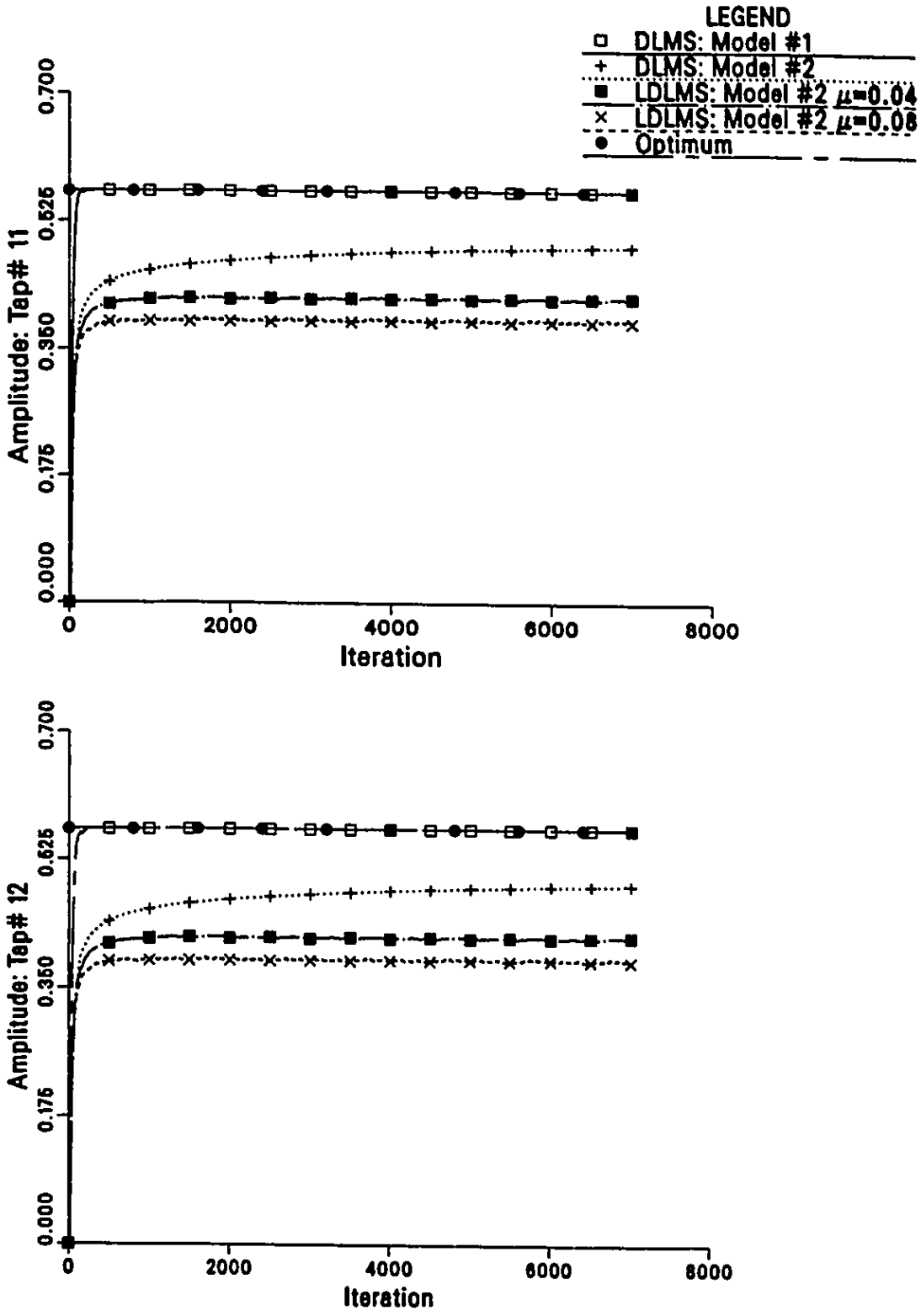


Figure 4.5: The DLMS and LDLMS mean tap-weight convergence using Model#1 and Model#2. Figures show the learning curves of the adaptive taps#11 and #12 for $D = 8$ and $\beta = 0.03$

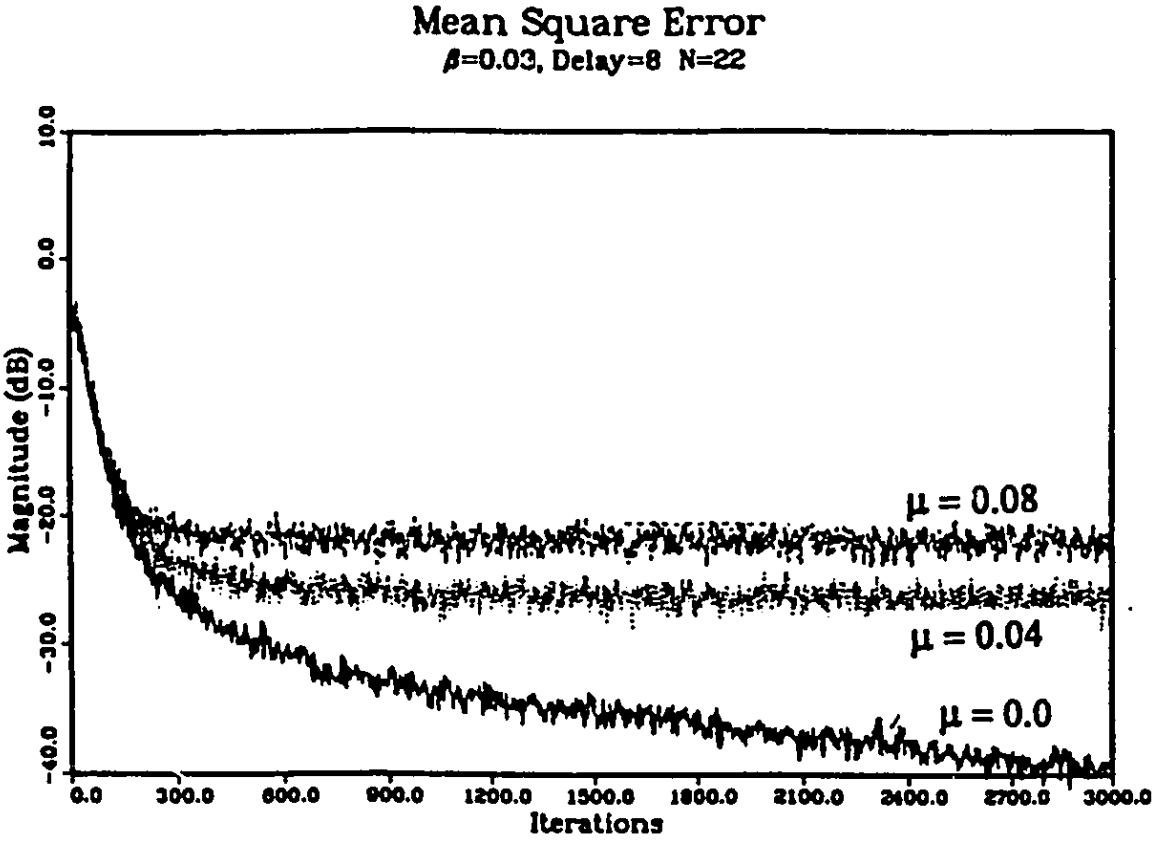


Figure 4.6: Learning curves of the LDLMS and DLMS algorithms using Model#1 and Model#2 for $D = 8$, μ=0, 0.04,0.08 and $\beta = 0.03$

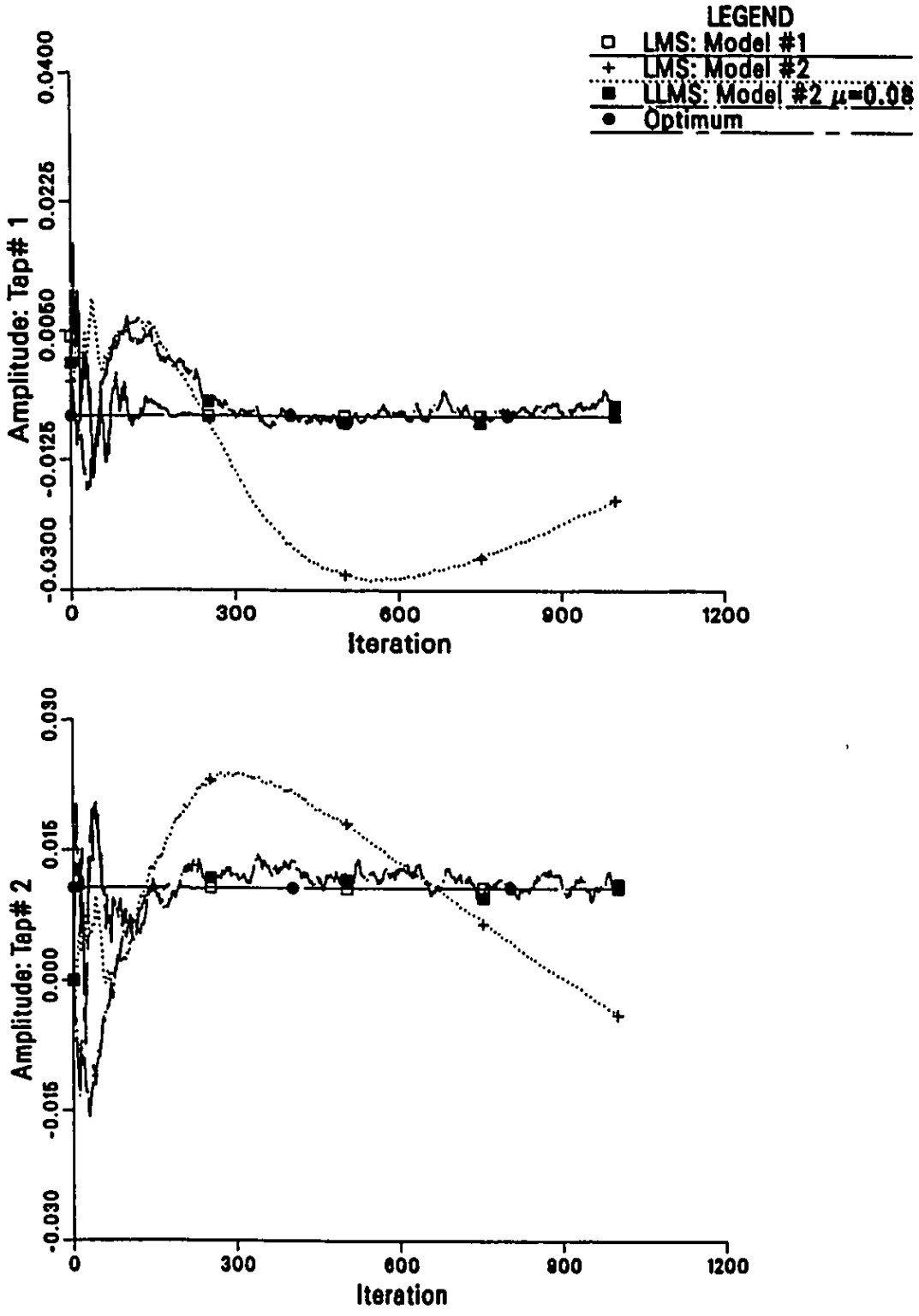


Figure 4.7: The LMS and LLMS mean tap-weight convergence using Model#1 and Model#2. Figures show the learning curves of the adaptive taps#1 and #2 for $\beta = 0.05$.

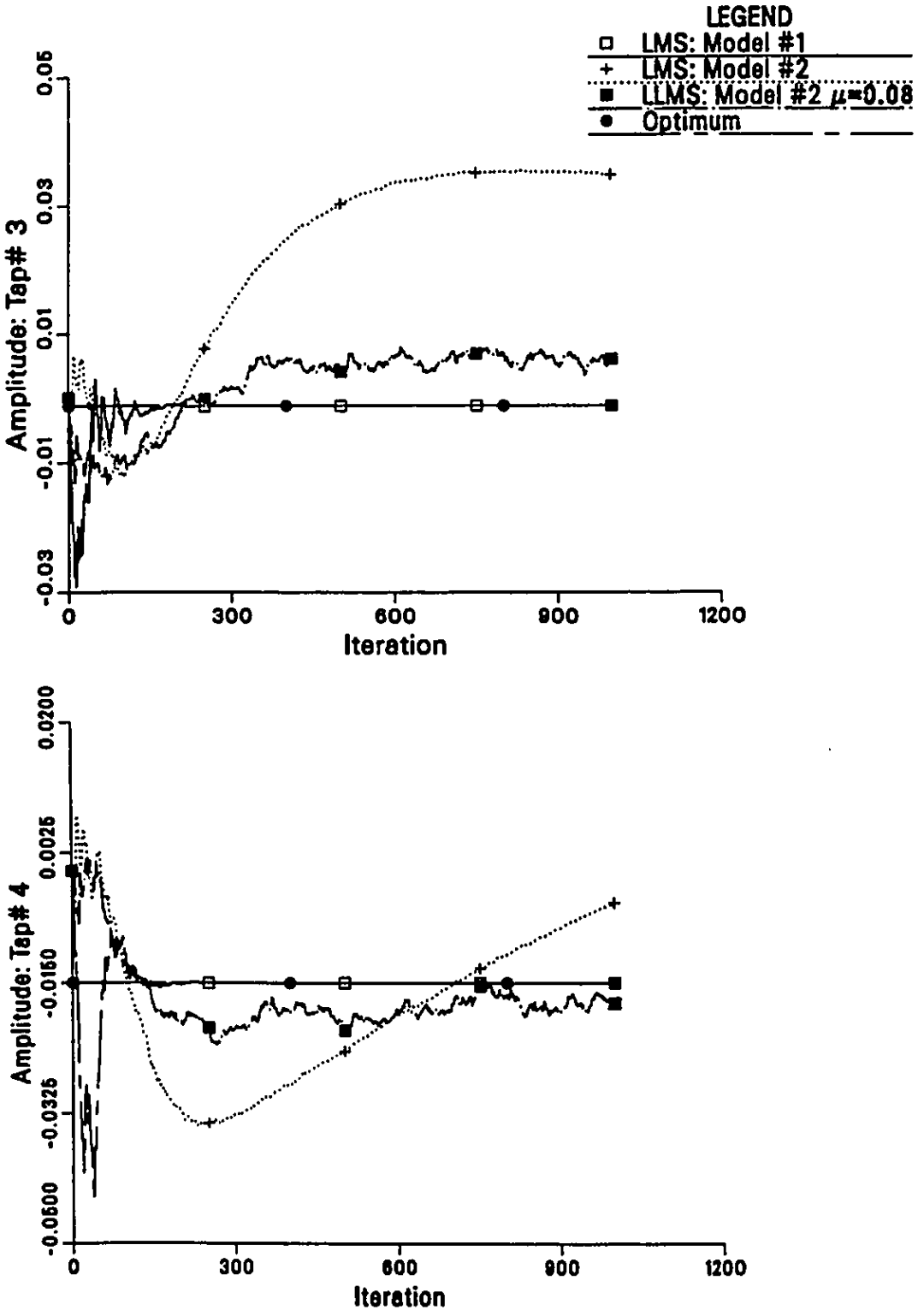


Figure 4.8: The LMS and LLMS mean tap-weight convergence using Model#1 and Model#2. Figures show the learning curves of the adaptive taps#3 and #4 for $\beta = 0.05$.

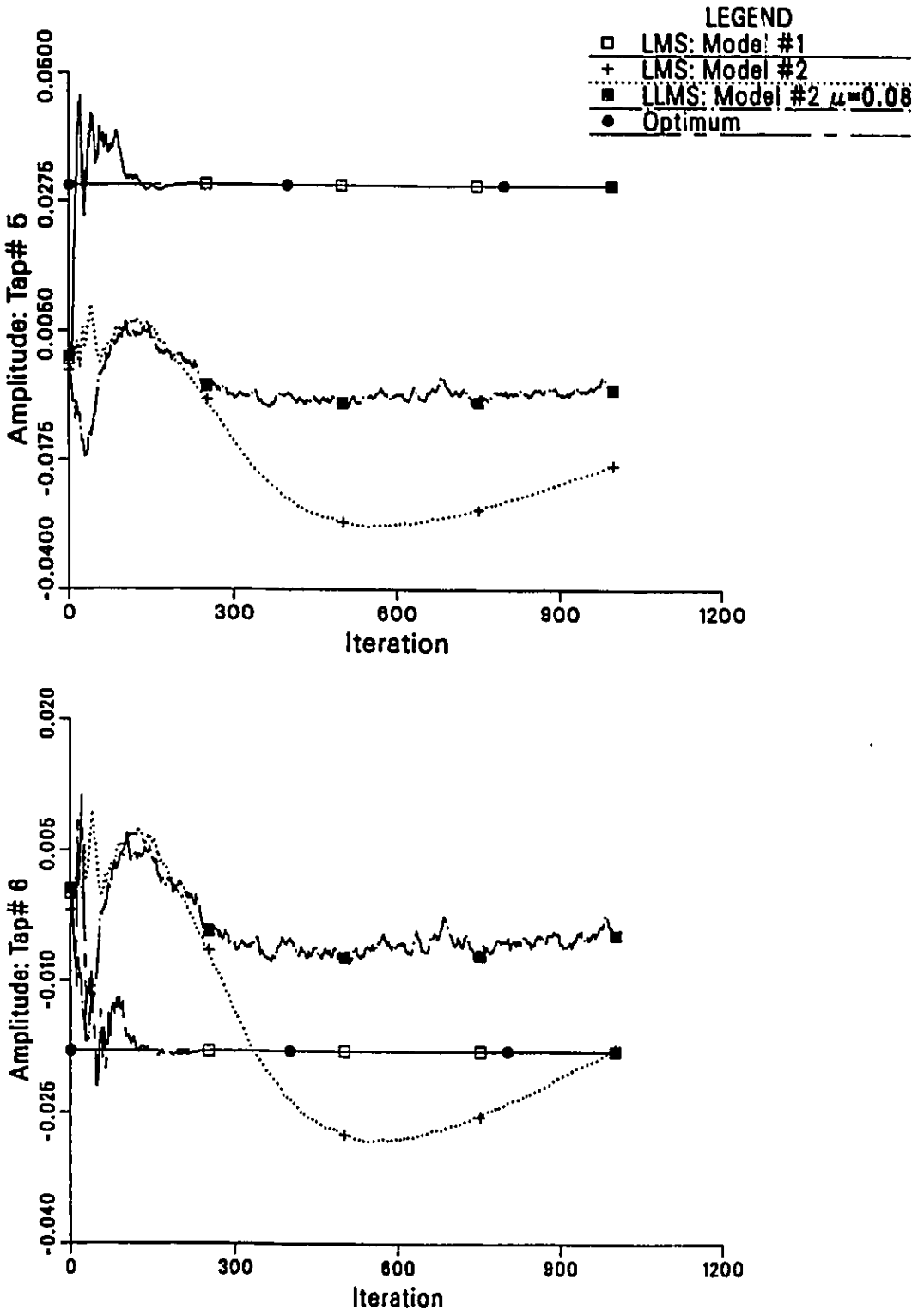


Figure 4.9: The LMS and LLMS mean tap-weight convergence using Model#1 and Model#2. Figures show the learning curves of the adaptive taps#5 and #6 for $\beta = 0.05$.

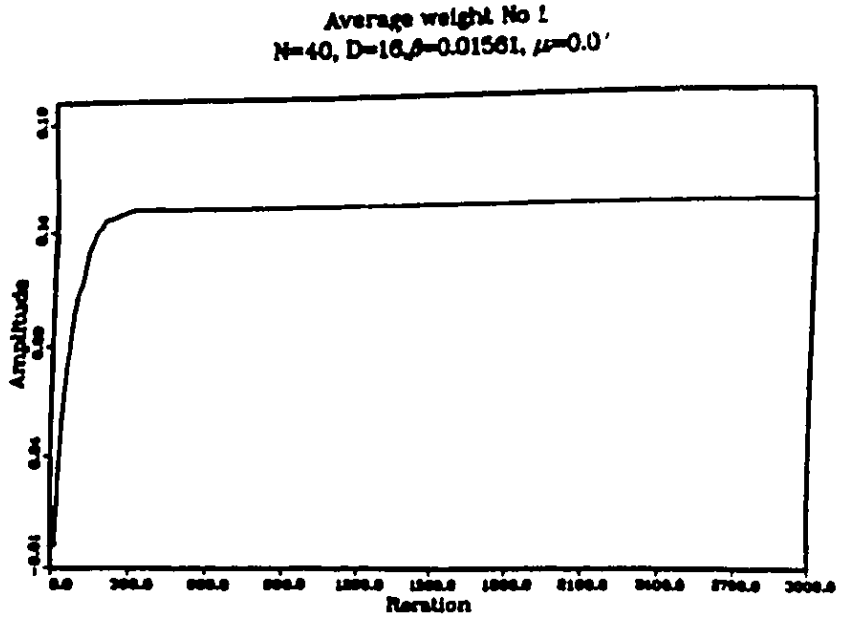


Figure 4.10: Stalling effects on the DLMS algorithm. When the gradient vanishes, the tap-weight is locked-up.

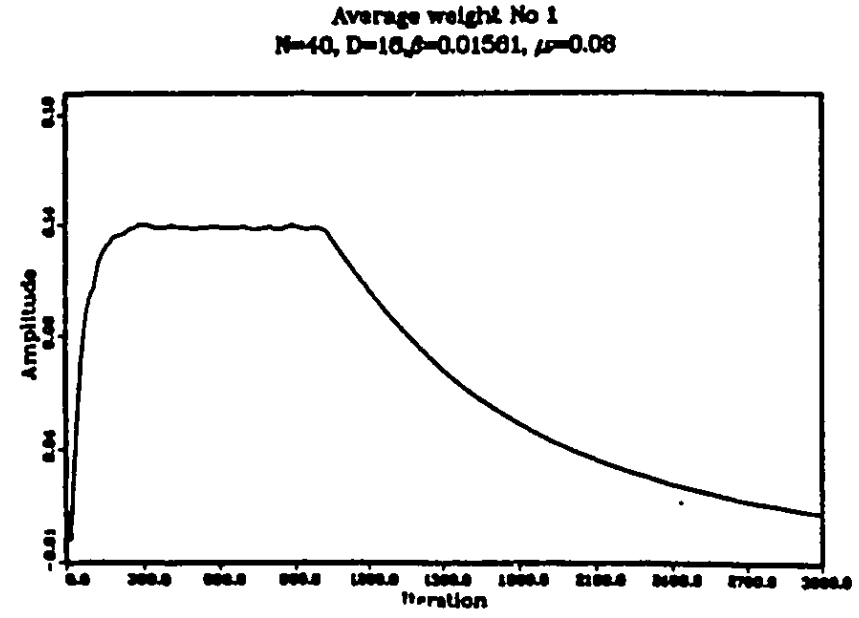


Figure 4.11: Effects of leakage tp prevent stalling in DLMS algorithm. The locked-up tap-weight converges or leaks to zero after using leakage.

no corresponding stimulus. This may drag the DLMS algorithm to undesirable results. Leakage in the DLMS algorithm will eventually prevent stalling from happening. This can be simply shown as follows. The LDLMS was defined in equation (3.3) and is rewritten here for convenience:

$$C(n) = (1 - \mu\beta)C(n-1) + \beta e(n-D)X(n-D). \tag{4.5}$$

If we let $\beta e(n-D)X(n-D) = 0$, we have the following:

$$C(n) = (1 - \mu\beta)C(n-1) \tag{4.6}$$

After l iterations, equation (4.6) can be written as

$$C(n+l) = (1 - \mu\beta)^l C(n-1) \tag{4.7}$$

If we let l grows without limit and since $(1 - \mu\beta)$ is slightly less than one,

$$\lim_{l \to \infty} C(n+l) = 0 \tag{4.8}$$

To illustrate this first order effect of this term, examples are shown in Figures 4.10- 4.11, where the input is white Gaussian noise. In this figure, we see that before using leakage, the tap-weight $c_1(n)$ is locked-up to a constant value when the input is very small or zero. This value is not the optimum but it results when the second term of (4.5) vanishes. After using the leakage μ, the locked-up coefficients go or leak to zero.

A step size β is selected from the stability bound derived earlier. The effect of zero and non-zero leakage on the coefficient convergence of the DLMS algorithm is shown for the average tap-weight $< c_1(n) >$. The gradient term in the update equation of the LDLMS algorithm was made very small to cause the stalling of the filter coefficients (locked up coefficients). In this example, the optimum value of $c_1(n)$ cannot be reached for both zero or non-zero leakage value. However, whenever the gradient value in the update equation becomes significant, $c_1(n)$ reconverges to its optimum value.

4.3 Application of the LDLMS algorithm in Non-stationary Environment

Adaptive filtering in nonstationary environments represent a major area in adaptive techniques. Both cost of implementation and performance determine the choice of an adaptive algorithm for a given application. The LMS algorithm is the usual choice for providing a good balance between cost and performance. The tracking performance of the LMS algorithm under time-varying systems was studied in [1] and [9].

The DLMS algorithm, however, does not perform as LMS because of the inherent delay incorporated in the LMS update equation. The DLMS algorithm was found to fail under nonstationary environments [2] and [4] relatively for large delays. In the following examples, the DLMS tracking capability is investigated for a particular type of a time-varying system identification shown in Figure 4.12. A performance measure defined here as the time-average misadjustment of the algorithms is employed for comparing the LMS, LLMS, DLMS, and LDLMS algorithms. It will be shown that leakage in both LMS and DLMS algorithms reduces the final excess MSE in nonstationary environments particularly when the step size is close to its upper bound. Moreover, leakage enhances the stability of the adaptive algorithm.

In nonstationary environments, the optimum tap-weight vector $C_{opt}(n)$ changes from one iteration to the next. So, the adaptive algorithm always tries to track continuously the moving position of the minimum point of its error performance surface. Also, the input correlation matrix H is fixed since it is assumed that the inputs are stationary. However, the desired signal $d(n)$ is made nonstationary thus the cross-correlation vector of the tap-inputs $P(n)$ assumes time varying form. The tap-weight error vector noise and tap-weight error vector lag result from this nonstationary situation. These account for two misadjustments due to lag of tracking and the second is due to the gradient noise. The former is inversely proportional to the step size β while the latter is directly proportional to β. Therefore, the optimum choice of β results when the sum of these two misadjustments is minimum [9]. The total misadjustment can be described by the

following expression:

$$Total\ Misadjustment = \frac{< (C(n) - C^*(n))^T H(C(n) - C^*(n)) >}{J_{min}} \tag{4.9}$$

where $C^*(n)$ is the tap-weight of the unknown system.

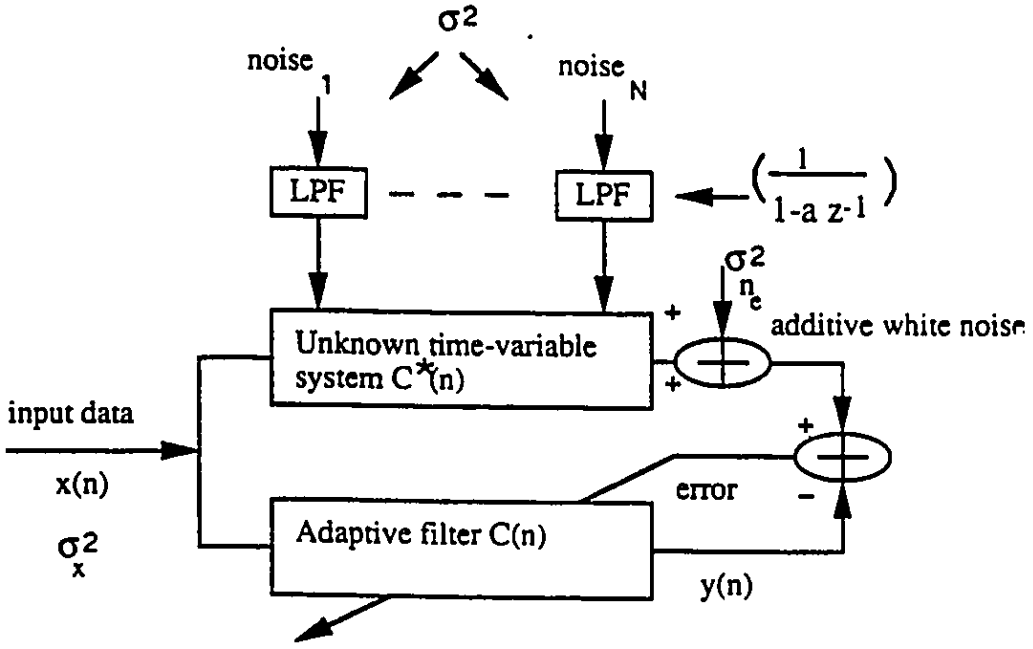


Figure 4.12: The unknown time-varying system identification model

4.3.1 Description of the Simulation Parameters

The task here involves identifying an unknown time-variable system. This unknown system and the adaptive filter both have the same filter length. The time-average misadjustment was measured for various step size values. An ensemble of 300 statistically independent input samples of the input $x(n)$ and the noise $n_e(n)$ with both unit variance white Gaussian noise was averaged over. The time-average excess MSE $J_{ex}(n)$ was averaged over different time samples after the initial transients had died away. The N tap-weight unknown parameters undergo an independent Gauss-Markov process and each

of the N tap-weight sequence is held fixed throughout the ensemble of excitation of the input and the noise. The first order Markov time series is given by:

$$C^*(n+1) = aC^*(n) + N_s(n). \tag{4.10}$$

where the parameter a denotes the degree of nonstationarity (DNS) of the time-varying system and $N_s(n)$ is the $N \times 1$ vector input noise sequences with each variance σ^2. Each noise sequence of $N_s(n)$ is statistically independent Gaussian noise. The simulations are carried under different signal to noise ratio (SNR), which is defined here as

$$SNR = \frac{N\sigma_x^2\sigma_c^2}{\sigma_{n_e}^2}, \tag{4.11}$$

where σ_c^2 is the variance of the $c_i^*(n)$, $i = 0, N-1$ sequences, which fluctuate independently of each other according to the first order-Markov process given in (4.10). J_{min} is in general the minimum mean-squared error of the adaptive algorithm when $C_{opt}(n) = C^*(n)$. The desired signal $d(n)$ is written as

$$d(n) = C^{*T}(n)X(n) + n_e(n). \tag{4.12}$$

When $C_{opt}(n) = C^*(n)$, $e_{opt}(n) = n_e(n)$, and hence $< e_{opt}^2(n) >= J_{min} = \sigma_{ne}^2$. The steady state variance of $C^*(n)$ is related to the variance of the low pass filtered input noise σ^2 as it is shown in Figure 4.12 by

$$\sigma_c^2 = \frac{\sigma^2}{1-a^2}. \tag{4.13}$$

4.3.2 Simulation Results

The effect of the delay upon the performance measure of the LMS algorithm is shown in Figures 4.13 and 4.14. We see that DLMS fails to behave as LMS algorithm and it has very high time-average misadjustment above step size $\beta = 0.04$. Because of the delay incorporated in the LMS algorithm, instability of the MSE appear above $\beta = 0.04$, giving very large time-average misadjustments. These values are not shown in the figures. The time-average misadjustment with SNR=10 in Figure 4.14 is higher than in the case of

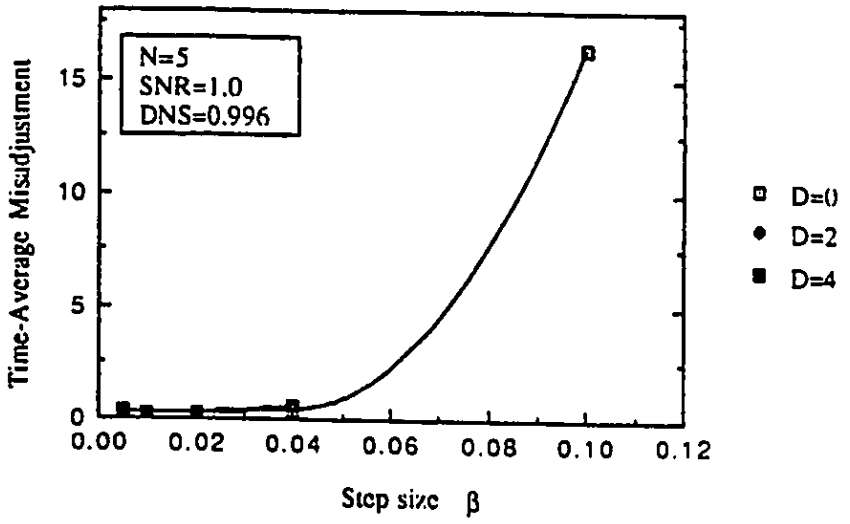


Figure 4.13: Time-average misadjustment vs step size. Comparison of time-average misadjustment between LMS and DLMS algorithms for SNR=1.

SNR=1.0 shown in Figure 4.13. Large values of the filter length N will also give larger time-average misadjustments than in the previous case for $N = 5$.

On the other hand, the use of leakage in DLMS improves the stability of the MSE and then improves the performance measure as it is depicted in Figures 4.15- 4.16. Furthermore, in Figure 4.15, the step size now can be extended up to $\beta = 0.1$ for the case of delay $D = 2$. Beyond this value, a very high misadjustment occurs.

To further reduce the final misadjustment, larger values of leakage μ should be employed. In this case, the performance measure is improved considerably and the algorithm is more stable than before. This is well illustrated in Figure 4.16 and 4.17.

The time-average misadjustment as a function of leakage is shown in Figure 4.18 and Figure 4.19 for LLMS and LDLMS algorithms. For step size value $\beta = 0.1$, the LMS and DLMS algorithms have large misadjustments. However, it can be shown in Figures 4.18 and 4.19 that as leakage increases, the system becomes more and more stable and the final misadjustments become smaller. Leakage can make the DLMS algorithm as well as the LMS algorithm robust and it improves the steady state misadjustments in nonstationary environments. In other words, with leakage in the algorithms, trackability of the unknown

system parameters can be improved since the leakage effectively makes the system forget the past optimum.

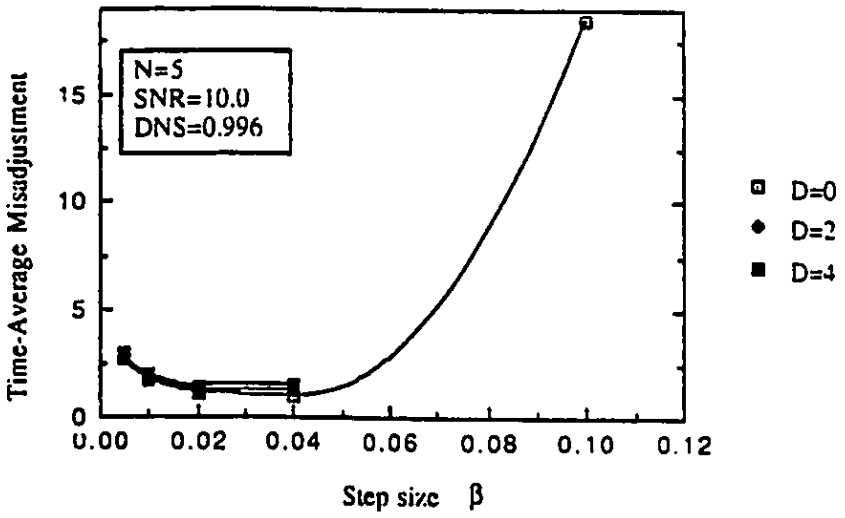


Figure 4.14: Time-average misadjustment vs step size. Comparison of time-average misadjustment between LMS and DLMS algorithms for SNR=10.

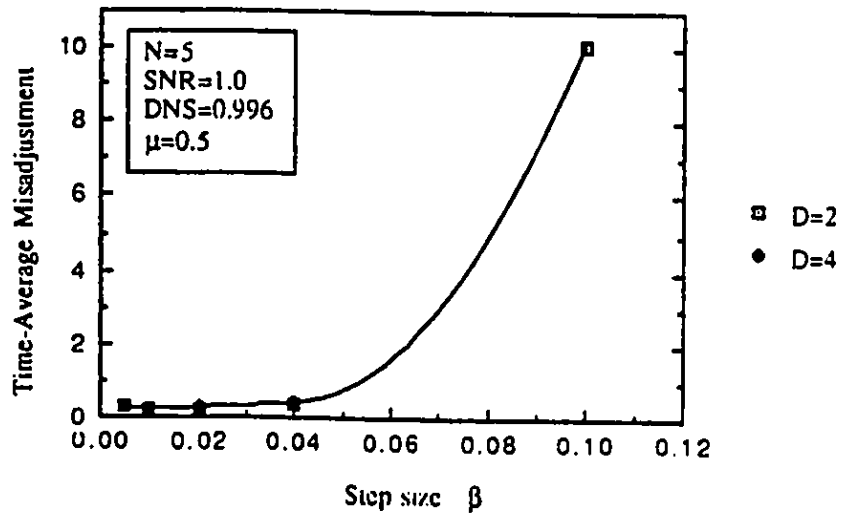


Figure 4.15: LDLMS time-average misadjustment vs step size for SNR=1 with $\mu = 0.5$.

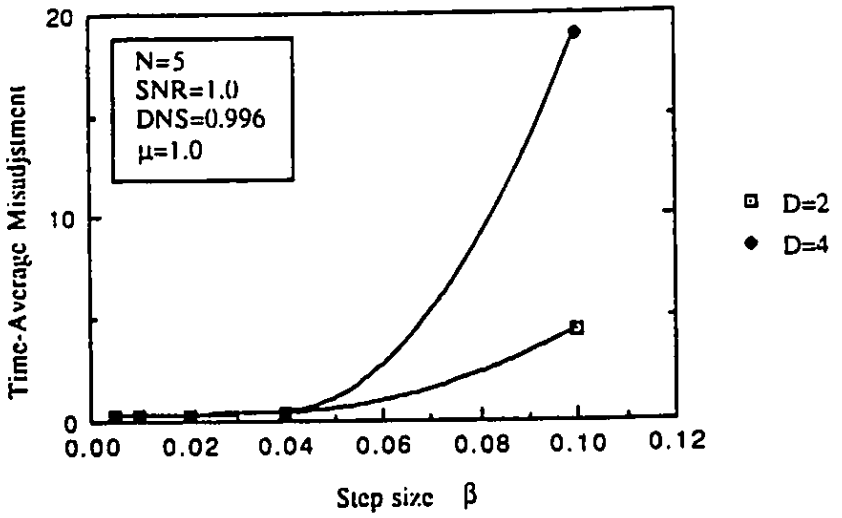


Figure 4.16: LDLMS time-average misadjustment vs step size for SNR=1 with $\mu = 1.0$.

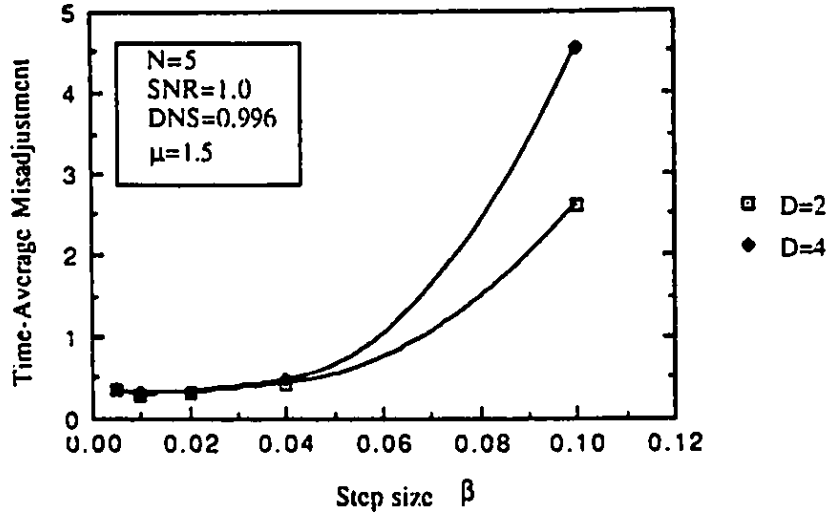


Figure 4.17: LDLMS time-average misadjustment vs step size for SNR=1 with $\mu = 1.5$.

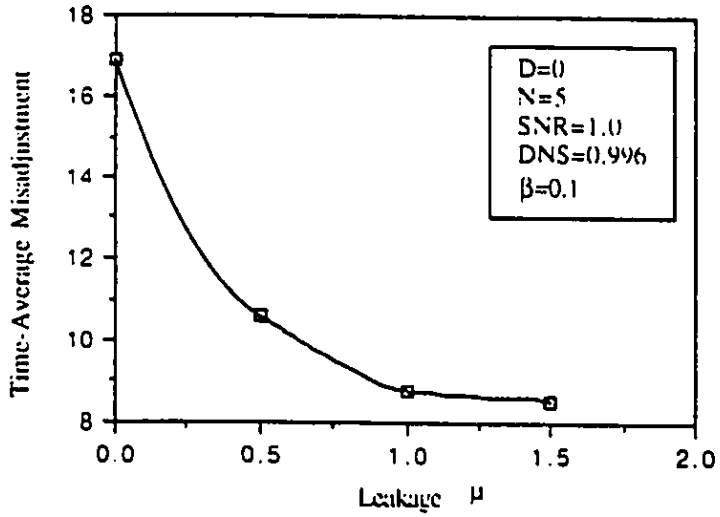


Figure 4.18: LLMS time-average misadjustment vs leakage for SNR=1.

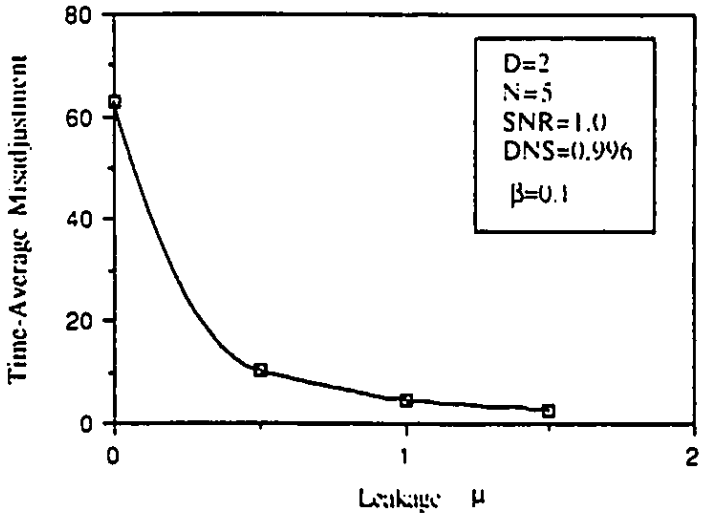


Figure 4.19: LDLMS time-average misadjustment vs leakage for SNR=1.

4.4 Conclusion

As an example of the application of LDLMS algorithm, the case of bandlimited input signal is considered. It was seen that the adaptive tap-weights are driven and undamped when the input fails to excite some or all the modes of the DLMS algorithm. Thus, the autocorrelation matrix H associated with the input process has one or more negligible or zero eigenvalues. The unsatbility of the DLMS filter coefficients in Model# 2 was clearly shown. On the other hand, the use of the LDLMS algorithm lead to stability and convergence of all filter coefficients. Two values of leakage were used and both prevented the filter coefficients to drift away and in undesirable manner from their optimum settings. The LDLMS acts as injecting a small amount of white noise into the filter input to quieten the coefficients. This is in fact a preferred method as compared to adding noise explicitly to the input process. In practice, the generation of this extra noise to be added at the input process of the adaptive filter may often be very difficult [22].

It was also shown that introduction of leakage speeds up convergence considerably, as compared to the DLMS, towards a suboptimum, which is in most case close to the true optimum. Despite the fact that the filter tap-weights diverge, the MSE, however, does converge as long as the step size is within the derived stability bounds. The eigenvalues of the input autocorrelation matrix determines the degree of penalty incurred in the cost function $J(n)$ used. For very small or negligible eigenvalues of H, the MSE converges to a very small value or close to its J_{min}.

A very small leakage is enough to counteract the undriven modes of the adaptive filter. Consequently, this small value of leakage introduces only a very small degradation on the steady state MSE.

Simulation examples for the case of the LMS and LLMS under both ideal and non-ideal input conditions were also illustrated. It was seen that the same behavior in the case of DLMS and LDLMS exist as well for the LMS and LLMS algorithms. Thus, previous

conclusions can be similarly applied here. For the second example, leakage was employed to avoid the stalling situation in the adaptive algorithms. This might be a desired case when it is preferable to have the tap-weights return to zero instead.

In the third application, the LLMS and LDLMS algorithms have been shown to perform better than the LMS and DLMS algorithms, respectively, in nonstationary environments. The leakage bounds the norm of the tap-weight error vector $\epsilon(n)$. Therefore, it stabilizes the system and reduces the steady state excess MSE of the algorithms considerably. In addition, it can be seen as a forgetting factor in the update equation of the algorithms so only small amount of information is used to track the time-varying parameters of the unknown system. The delay has detrimental effects upon the tracking capability of the LMS, especially for large delay and the use of leakage can be beneficial in improving stability and reducing the performance measure.

Chapter 5

Summary and Conclusions

In this thesis, we introduced a thorough study and analysis of the leaky LMS algorithm with delayed adjustment in stationary environment for a system identification model. Stability and convergence of the leaky DLMS algorithm were examined in two different ways. In the first approach, stability analysis was based on the convergence of the filter coefficient of the adaptive filter. The root locus technique was employed to illustrate the behavior of the DLMS algorithm with and without leakage when subjected to unstability. The Z-transform of the update equation of the leaky DLMS algorithm was taken to investigate the locations of the poles. It has been seen how one or more eigenvalues can easily drive the DLMS algorithm's poles outside the unit circle. Also, a step size which is very close to its upperbound can lead to unstability problems. However, when leakage is introduced in the DLMS and stability problems in both cases are prevented. It was seen that a small value of the leakage in the DLMS algorithm is sufficient to counteract instability. Consequently, this amount of leakage introduces only a small value of bias on the true optimum weight vector C_{opt}. In addition, a comparison of stability bounds of other well known algorithms namely LMS, leaky LMS, and delayed LMS algorithms was made. The upper bound on the step size of the leaky DLMS algorithm was shown to be much tighter than the other ones. Furthermore, the derived stability bounds of the LDLMS on the mean sense was seen to depend on the eigenvalues of the input autocorrelation matrix, leakage parameter, and the delay. The leaky DLMS characteristic equation has

complex roots; which is not the case for the LMS and leaky LMS algorithms. As long as the amplitude of the real root of the characteristic equation dominates, no degradation on the performance of the algorithm appears. However, when the delay values become too large, more complex roots start dominating and thus unacceptable degradation the leaky DLMS algorithm results.

In the second case, the stability of the LDLMS algorithm was based on the convergence of the mean square error. An approximation of the steady state excess mean-squared error was obtained as a function of filter length, input signal power, eigenvalue distribution, leakage parameter, delay, and kurtosis as well as the step size. This is a general expression from which the steady state excess MSE and the stability bounds of other well known algorithms LMS, LLMS and DLMS can be obtained. Approximate stability bounds were obtained as a function of the LDLMS step size. The accuracy of the approximation was found to depend on the ratio $\frac{D}{N}$. When $\frac{D}{N}$ is very close to one, the accuracy of these simple opproximations is not acceptable. Thus, high order approximations have to be found by using a computer. The $J_{ex}(\infty)$ was found to depend on the leakage parameter. With appropriate leakage value, the $J_{ex}(\infty)$ can be made insensitive to μ. Consequently a small extra excess MSE can be tolerated.

The theoretical results were verified by computer simulations and close agreement was obtained. The difference between these two results is first due to the gradient noise since we are not using exact statistics of the inputs. Second, the difference also depends on the assumptions and approximations made throughout the theoretical analysis of the algorithm.

Several applications of the leaky DLMS algorithm were considered to study the performance of the algorithm under non-ideal situations. First, an input with insufficient spectral content was considered. Even though the step size was within the derived stability bound, the DLMS algorithm's filter tap-weights drift away from the optimum settings. On the other hand, inserting leakage in the DLMS update equation, all the tap-weights are driven to their optimum with a small amount of bias. It was seen that the error converges despite of the weak input excitations. This is similar to the zero delay case.

Other applications of the leaky DLMS algorithm were also introduced. In the example of the stalling phenomenon, we showed that leakage returns the locked up adaptive filter coefficients to zero, which is a situation that might be preferred in some applications. In the last application, the tracking capabilities of both LMS and DLMS algorithms were examined under nonstationary environments with zero and non-zero leakage. It was shown that leakage stabilizes the system and reduces the steady state MSE.

The analysis of the LDLMS algorithm so far presented in this thesis can be as well applied to any applications of adaptive algorithms and is not restricted to the system identification case.

Finally, few questions still need to be addressed.

- Performance of the DLMS algorithm when implemented with finite precision needs to be studied. This topic is of special relevance here because the step size of the DLMS algorithm is much smaller than the LMS algorithm.

- The same can be repeated for LDLMS to verify its ability to improve performance under non-ideal situations even when implemented in finite precision.

- In some applications, the sign-error LMS algorithm with a feedback delay can also be a good candidate to the DLMS. In that case, the effect of delay upon the performance of the sign-error LMS algorithm should be examined.

- The usefulness of leakage sign input/error LMS should be investigated with and without delay.

- Comparison between these two algorithms should be made.

Appendix A

Equation Derivation for the LDLMS Algorithm

Here we derive an expression for $a_{D,D}(n-1)$ in terms of delays and leakage parameter. First, for simplicity, let $\phi = \mu\beta$, $\gamma = 1-\phi$, and $b = \beta\sigma^2$. Then, we define

$$a_{D,k}(m) =< W_m^T \Lambda W_{m-k} > \tag{A.1}$$

With the use of equation (3.38), we can get

$$\begin{aligned}
a_{D,k}(m) &= < W_m^T \Lambda W_{m-k} > \\
&= < (\gamma W_{m-1}^T + \beta e_o W_{m-D-1} Z_{m-D} Z_{m-D}^T - \phi R)\Lambda W_{m-k} > \\
&\approx \gamma a_{D,k}(m-1) - b < W_{m-D-1}^T \Lambda W_{m-k} > -\phi < R^T \Lambda W_{m-k} > \\
&\approx \cdots \\
&\vdots \\
&\approx \gamma^k a_{D,0}(m-k) - b \sum_{j=D-k+1}^{D} \gamma^{j-D+k-1} a_{D,j}(m-k) \\
&\quad -\phi < R^T \Lambda W_{m-k} > \sum_{j=1}^{k} \gamma^{j-1}
\end{aligned} \tag{A.2}$$

Let $k = D$ and $m = n-1$, we get

$$a_{D,D}(n-1) = \gamma^D a_{D,0}(n-D-1) - b\sum_{j=1}^{D} \gamma^{j-1} a_{D,j}(n-D-1)$$

$$-\phi < R^T \Lambda W_{n-D-1} > \sum_{j=1}^{D} \gamma^{j-1} \tag{A.3}$$

Let's denoted $\psi(\gamma)$ by

$$\psi(\gamma) \;=\; \phi < R_o^T \Lambda W_{n-D-1} > \sum_{j=1}^{D} \gamma^{j-1} \tag{A.4}$$

We note that for D=0, $a_{0,0}(n-1) = J_{ex}(n-1)$. Now, iterating (A.3) again, we find

$$\begin{aligned}
a_{D,D}(n-1) \;=\; & \gamma^D a_{D,0}(n-D-1) - b\sum_{j=1}^{D}\gamma^{2j-1} a_{D,0}(n-D-j-1) \\
& + b^2 \sum_{j=1}^{D}\gamma^{j-1} \sum_{i=D-j+1}^{D} \gamma^{2i-1} a_{D,0}(n-D-j-i-1) \\
& - b^3 \sum_{j=1}^{D}\gamma^{j-1} \sum_{i=D-j+1}^{D} \gamma^{i-1} \sum_{s=D-i+1}^{D} \gamma^{2s-1} a_{D,0}(n-D-j-i-s-1) \\
& + b^4 \sum_{j=1}^{D}\gamma^{j-1} \sum_{i=D-j+1}^{D} \gamma^{i-1} \sum_{s=D-i+1}^{D} \gamma^{s-1} \sum_{r=D-s-1}^{D} \gamma^{r-1} a_{D,0}(n-D-j-i-s-r-1) \\
& - \psi(\gamma) + b\psi(\gamma)\sum_{j=1}^{D}\gamma^{j-1} - b^2\phi\sum_{j=1}^{D}\gamma^{j-1} \sum_{i=D-j+1}^{D} \gamma^{i-1} \\
& + b^3\psi(\gamma)\sum_{j=1}^{D}\gamma^{j-1} \sum_{i=D-j+1}^{D} \gamma^{i-1} \sum_{s=D-i+1}^{D} \gamma^{s-1} + \cdots
\end{aligned} \tag{A.5}$$

We approximate the above expression for $\gamma \approx 1$ since this factor is between zero and one and when raised to a power, it becomes very small. Also, we use this to enable us to get closed forms for the sums in (A.5). The justification of these approximations are well verified by computer simulations. Consequently, Eq. (A.5) can be simplified as

$$\begin{aligned}
a_{D,D}(n-1) \;\approx\; & J_{ex}(n-D-1) - b\sum_{j=1}^{D} J_{ex}(n-D-j-1) \\
& + b^2 \sum_{j=1}^{D} \sum_{i=D-j+1}^{D} J_{ex}(n-D-j-i-1) \\
& - b^3 \sum_{j=1}^{D} \sum_{i=D-j+1}^{D} \sum_{s=D-i+1}^{D} J_{ex}(n-D-j-i-s-1) \\
& + b^4 \sum_{j=1}^{D} \sum_{i=D-j+1}^{D} \sum_{s=D-i+1}^{D} \sum_{r=D-s-1}^{D} J_{ex}(n-D-j-i-s-r-1)
\end{aligned}$$

$$
\begin{aligned}
&-\phi D < R_o^T \Lambda W_{n-D-1} > +b\phi D^2 < R_o^T \Lambda W_{n-D-1} > \\
&-b^2\phi \frac{D^2(D+1)}{2} < R_o^T \Lambda W_{n-D-1} > \\
&+b^3\phi \frac{D^2(D+1)(2D+1)}{2} < R_o^T \Lambda W_{n-D-1} > + \cdots
\end{aligned} \tag{A.6}
$$

In practical applications of adaptive filtering, the term bD is very small(i.e. $bD \ll 1$). In this case, the first term of the last two lines of equation (A.6) is kept. The other terms of these lines are discarded because they are very small when compared with the first term of the same lines. Hence, (A.6) con be finally simplified as:

$$
\begin{aligned}
a_{D,D}(n-1) \approx\ & J_{ex}(n-D-1) - b\sum_{j=1}^{D} J_{ex}(n-D-j-1) \\
&+b^2\sum_{j=1}^{D}\sum_{i=D-j+1}^{D} J_{ex}(n-D-j-i-1) \\
&-b^3\sum_{j=1}^{D}\sum_{i=D-j+1}^{D}\sum_{s=D-i+1}^{D} J_{ex}(n-D-j-i-s-1) \\
&+b^4\sum_{j=1}^{D}\sum_{i=D-j+1}^{D}\sum_{s=D-i+1}^{D}\sum_{r=D-s-1}^{D} J_{ex}(n-D-j-i-s-r-1) \\
&-\phi D < R_o^T \Lambda W_{n-D-1} > + \cdots
\end{aligned} \tag{A.7}
$$

Appendix B

Derivation of the Tap-Weight Error Expression

If we consider the true statistics of the signal parameters, the L-LMS algorithm update equation can be given as the following:

$$C(n) = (\gamma I - \beta H)C(n-1) + \beta P \tag{B.1}$$

where for simplicity we denote $\gamma = 1 - \beta\mu$, and H and P are the correlation and the cross-correlation matrices, respectively. The tap-weight error vector is given by

$$\epsilon(n) = C(n) - C_o \tag{B.2}$$

With the substitution of the above equation into Eq. (B.1) and computing for $\epsilon(n+1)$, we obtain

$$\epsilon(n+1) = (\gamma - 1)C_o + (\gamma I - \beta H)\epsilon(n) \tag{B.3}$$

Let's utilize an iterative procedure for the above equation:

$$\begin{aligned}
\epsilon(1) &= (\gamma - 1)C_o + (\gamma I - \beta H)\epsilon(0) \\
\epsilon(2) &= (\gamma - 1)C_o + (\gamma I - \beta H)\epsilon(1) \\
&= (\gamma - 1)C_o + (\gamma I - \beta H)\left[(\gamma - 1)C_o + (\gamma I - \beta H)\epsilon(0)\right] \\
&= (\gamma - 1)C_o + (\gamma - 1)(\gamma I - \beta H)C_o + (\gamma I - \beta H)^2\epsilon(0)
\end{aligned}$$

$$\vdots$$

$$\epsilon(n) = (\gamma I - \beta H)^n \epsilon(0) + (\gamma - 1) \sum_{i=0}^{n-1} (\gamma I - \beta H)^i C_o \tag{B.4}$$

A well-known following expression [13] is

$$(I - \beta H)^n = \sum_{k=0}^{N-1} (1 - \beta \lambda_k)^n V_k V_k^T \tag{B.5}$$

However, for the case of the L-LMS algorithm, we have

$$(\gamma I - \beta H)^n = \sum_{k=0}^{N-1} (\gamma - \beta \lambda_k)^n V_k V_k^T \tag{B.6}$$

Using the above equation in Eq. (B.4), we obtain

$$\begin{aligned}
\epsilon(n) &= \sum_{k=0}^{N-1} (\gamma - \beta \lambda_k)^n V_k V_k^T \epsilon(0) + (\gamma - 1) \sum_{i=0}^{n-1} \sum_{k=0}^{N-1} (\gamma - \beta \lambda_k)^i V_k V_k^T C_o \\
&= \sum_{k=0}^{N-1} (1 - \beta \lambda_k)^n V_k^T \epsilon(0) V_k + (\gamma - 1) \sum_{k=0}^{N-1} \left[\sum_{i=0}^{n-1} (\gamma - \beta \lambda_k)^i \right] V_k V_k^T C_o \\
&= \sum_{k=0}^{N-1} (\gamma - \beta \lambda_k)^n V_k^T \epsilon(0) V_k + (\gamma - 1) \sum_{k=0}^{N-1} \frac{1 - (\gamma - \beta \lambda_k)^n}{1 - (\gamma - \beta \lambda_k)} C_o
\end{aligned} \tag{B.7}$$

In order to have a clear view of the modified eigenvlaues in the above equation, we substitute the parameter μ into equation (B.7), then we reach the following expression:

$$\epsilon(n) = \sum_{k=0}^{N-1} (1 - \beta(\mu + \lambda_k))^n V_k^T \epsilon(0) V_k - \mu \sum_{k=0}^{N-1} \frac{1 - (1 - \beta(\mu + \lambda_k))^n}{\mu + \lambda_k} C_o \tag{B.8}$$

Appendix C

Filter Coefficients of the Finite Impulse Responses FIR1 and FIR2

C.1 Filter coefficients of FIR1

FIR1 has the following filter coefficients H:

H(1)=	7.8430180e-03	H(7)=	3.7380980e-01
H(2)=	-1.3320920e-02	H(8)=	1.8103030e-01
H(3)=	-4.9285890e-02	H(9)=	-3.9672850e-03
H(4)=	-3.9672850e-03	H(10)=	-4.9285890e-02
H(5)=	1.8103030e-01	H(12)=	7.8430180e-03
H(6)=	3.7380980e-01		

C.2 Filter coefficients of FIR2

FIR1 has the following filter coefficients H:

H(1)=	-6.6223140e-03	H(12)=	5.6610110e-01
H(2)=	1.0650630e-02	H(13)=	-3.2531740e-02
H(3)=	-1.3122560e-03	H(14)=	-8.3587650e-02
H(4)=	-1.4953610e-02	H(15)=	7.8857420e-02
H(5)=	3.0242920e-02	H(16)=	-2.5573730e-02
H(6)=	-1.7944340e-02	H(17)=	-1.7944340e-02
H(7)=	-2.5573730e-02	H(18)=	3.0242920e-02
H(8)=	7.8857420e-02	H(19)=	-1.4953610e-02
H(9)=	-8.3587650e-02	H(20)=	-1.3122560e-03
H(10)=	-3.2531740e-02	H(21)=	1.0650630e-02
H(11)=	5.6610110e-01	H(22)=	-6.6223140e-03

Bibliography

[1] B. Widrow et al. 'Stationary and Nonstationary Learning Characteristics of the LMS Adaptive Filter', Proc. IEEE, vol. 64, pp.1151-1162, Aug. 1976.

[2] Maurice G. Bellenger, Adaptive Digital Filters and Signal Processing, NY Marcell Dellker, 1987.

[3] G. Long et al., 'The LMS Algorithm with Delayed Coefficient Adaptation', IEEE Trans. on ASSP., Vol.37, no.9, Sept. 1989, pp.1397-1409.

[4] G. Long et al. ' Corrections of the paper:The LMS Algorithm with Delayed Coefficient adaptations', IEEE Trans. on ASSP., Vol 37, No. 9, pp. 1397-1405, Sept.1989.

[5] John R. Treichler et al., Theory and Design of Adaptive Digital Filters, New York: John Wiley & Sons, 1987.

[6] Peter Kabal, 'The Stability of Adaptive Minimum Mean Square Error Equalizers Using Delayed Adjustment', IEEE Trans. on Comm., Vol. Com31., No. 3, pp. 430-441, March 1983.

[7] R.D. Gitlin and S. B. Weinstein, 'On the Required Tap-Weight Precision for Digitally Implemented, Adaptive Mean-Squared Equalizers', BSTJ Vol. 58, No.2, pp. 301-321, Oct. 1982.

[8] R. P.Gitlin et al.,'The Tap-Leakage Algorithm: An Algorithm for the Stable Operation of a Digitally Implemented, Fractionally Spaced Adaptive Equalizer', BSTJ, Vol. 61, No. 8, pp. 1817-1839, Oct.1982.

[9] W. A. Gardner ,'Nonstationary Learning Characteristics of the LMS Algorithm', IEEE Trans. on CAS, Vol.CAS-34, NO.10, pp. 1199-1207, Oct.1987.

[10] G. Long et al., 'Study of a Pole-Zero Adaptive Echo Cancellers', IEEE Trans. on CAS, Vol. CAS-34, No.7, pp. 765-769, July 1987.

[11] Cohn D. L. and J. L. Melsa, 'The Residual Encoder:An Improved ADPCM Systems for Speech Digitization', IEEE Trans. on Comm., Vol. Com-23, pp.935-941, Sept. 1975.

[12] B. Widrow and S. D. Stearns, Adaptive Signal Processing, Englewood Cliffs, N.J.: Prentice- Hall, 1985.

[13] S. Haykin, Adaptive Filter Theory, Englewood Cliffs, N.J. : Prentice Hall, 1986.

[14] C. C. Evci et al., 'Characteristics of Adaptive Filters With Leakage', Proceedings of ICASSP, pp. 3.9.1-3.9.4, 1984.

[15] T. Kailath and J.M. Cioffi, 'Fast Recursive-Least Squares Transversal Filter for Adaptive Filtering', IEEE Trans. on ASSP, Vol. ASSP-32, No.2, pp.304-337, April 1984.

[16] D.D. Falconer and L. Ljung ' Applications of Fast Kalman Estimation to Adaptive Equalization', IEEE Trans. on Comm., Vol. Comm-26, No.10, Oct. 1978.

[17] J. M. Cioffi, Fast Transversal Filters for Communication Applications, Phd. Dissertation, Stanford Univ., Stanford, C.A, March 1983.

[18] S. H. Alardan, L.J.Faber, 'A Fast ARMA Transversal RLS Filter Algorithm', IEEE Trans. ASSP Vol. 36, No.3, pp.349-358, March 1988.

[19] C. R. Johnson, Jr., Lectures on Adaptive Parameter Estimation. Englewood Cliffs N.J.: Prentice-Hall, 1988.

[20] C. R.Johnson et al. 'Excitation Conditions for Signal Regressor Least Mean Squares Adaptation', IEEE Trans. on CAS, Vol. 35, No.6, June 1988.

[21] V.J. Mathews and S.H. Cho, 'Improved Convergence Analysis of Stochastic Gradient Adaptive Filters Using the Sign Algorithm', IEEE Trans. ASSP, Vol. ASSP-35, No.4, pp.450-454, April 1987.

[22] John M. Cioffi, 'Limited-Precision Effects in Adaptive Filtering', IEEE Trans. on CAS., Vol-CAS-34, No.7, pp.821-833, July 1987.

[23] S. Dasgupta et al., 'Comparison between Leaky Sign-Sign and LMS Adaptive Algorithms', IFAC Identification and System Parameter Estimation, Beijing, pp.311-314, PRC 1988.

[24] H.Kobayashi, 'Correlative Level Coding and Maximum Likelihood Decoding',IEEE Trans. Inform. Theory, Vol. IT-17, pp.586-594, Sept. 1971.

[25] G.D.Forrey, Jr. 'Maximum-Likelihood Sequence Estimation of Digital Sequences in the Presence of Intersymbol Interference', IEEE Trans. Inform. Theory, Vol. IT-18, pp. 363-378, May 1972.

[26] N.S. Jayant, Digital Coding of Waveforms:Principles and Applications to Speech and Video, Prentice Hall, Englewood Cliffs, N.J., 1984.

[27] W. A. Gardner, ' Learning Characteristics of Stochastic Gradient Descent Algorithms: A general study analysis, and critique',signal processing, Vol.6, pp.113-133, 1984.

[28] Peter J. Voltz, 'Sample Convergence of the Normed LMS Algorithm With Feedback Delay', ICASSP, pp.2129-2132, 1991.

[29] Alan Weiss and Debasis Mitra, 'Digital Adaptive Filters: Conditions for Convergence, Rates of Convergence, Effects of Noise and Errors Arising From Implementation', IEEE Trans. On information Theory, Vol.IT-25, No.6, Nov. 1979.

[30] N. J. Bershad, 'Analysis of the Normalized LMS Algorithm With Gaussian Inputs', IEEE Trans. ASSP, Vol. ASSP-34, pp.793-806, Aug. 1986.

[31] N. J. Bershd, 'Behavior of the ϵ-Normalized LMS Algorithm With Gaussian Inputs', IEEE Trans. ASSP, Vol. ASSP-35, pp.636-644, May 1987.

[32] D. D. Falconer, 'Adaptive Reference Echo Cancellation', IEEE Trans. on Commun., vol. Com-30, No.9, Sept. 1982.

[33] J. G. Proakis, Digital Communications, 2nd edition, McGraw Hill Book Company, 1989.

[34] Martin D. Meyer and Dharma P. Agrawad, ' An Arbitrarily High Sampling Rate DLMS Adaptive Filter.', IEEE Trans. On Commun., Vol.69, No. 1, pp.2176 2179, January 1992.

[35] V. B. Lawrence ' Multiprocessor Implementation of Adaptive Filters', IEEE Trans. on Commun., Vol. Com31, No.6, pp. 826-835, June 1983.

[36] William A. S. et al. 'Parameter Drift in LMS Adaptive Filter'; IEEE Trans. on ASSP, Vol. ASSP-34, No. 4, pp. 868-878, August 1986.