Recursive Inverse Adaptive Filtering Techniques And Applications

Mohammad Mustafa Shukri AHMAD

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical and Electronic Engineering

Eastern Mediterranean University
September 2011
Gazimagusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Elvan Yılmaz
Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Doctor of Philosophy in Electrical and Electronic Engineering

Assoc. Prof. Dr. Aykut Hocanın
Chair of
Electrical and Electronic Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality as a thesis of the degree of Doctor of Philosophy in Electrical and Electronic Engineering

Prof. Dr. Osman Kükrer
Cosupervisor

Assoc. Prof. Dr. Aykut Hocanın
Supervisor

Examining Committee

1. Prof. Dr. A. Enis Çetin
2. Prof. Dr. Osman Kükrer
3. Prof. Dr. Runyi Yu
4. Assoc. Prof. Dr. Aykut Hocanın
5. Assoc. Prof. Dr. Erhan İnce

ABSTRACT

Adaptive filtering techniques are widely used to cope with the variations of system parameters. In finite impulse response (FIR) adaptive filtering, the filter weights are updated iteratively by minimizing the mean-square-error (MSE) of the difference between the desired response of the adaptive filter and its output. However, most of the existing adaptive filters experience many difficulties; fixed-step size which provides poor performance in highly correlated environments, high computational complexity, stability due to the inversion of the autocorrelation matrix, tracking ability in non-stationary and impulsive noise environments.

The novelty of this work resides in the derivation of a new FIR recursive inverse (RI) adaptive filtering algorithm. This algorithm has been proposed to overcome some of the difficulties experienced with the existing adaptive filtering techniques. The approach uses a variable step-size and the instantaneous value of the autocorrelation matrix in the coefficient update equation that leads to an improved performance. Avoiding the use of the inverse autocorrelation matrix, as the case of the recursive-least-squares (RLS) algorithm, would provide more stable performance. Convergence analysis of the algorithm has been presented. The ensemble-average learning curve of the RI algorithm is derived and compared with those of the RLS and least-mean-square (LMS) algorithms. A general fast implementation technique, which significantly reduces the computational complexity, of the RI algorithm is presented. A robust version of the RI algorithm, which leads to an improved performance, in impulsive noise environments is presented. The effect of the forgetting factor on the performance of the RI algorithm is investigated. Also, a two-

dimensional (2D) version of the RI algorithm is introduced. Finally, a second-order version of the RI algorithm, which provides further improvement in the performance, is derived.

The performance of the RI, fast RI, proposed robust RI (PRI), second order RI and 2D RI algorithms is compared to those of the standard LMS, normalized LMS (NLMS) , variable step size LMS (VSSLMS), discrete cosine transform LMS (DCT-LMS), transform domain LMS with variable step-size (TDVSS), RLS, stabilized fast transversal RLS (SFTRLS), robust RLS (RRLS) and proposed robust RLS algorithms in additive white Gaussian noise (AWGN), correlated Gaussian noise and white and correlated impulsive noise, in noise cancellation, system identification, channel equalization, echo cancellation and image deconvolution setting, in stationary and non-stationary environments. Simulations show that the RI algorithm and its variants outperform all the aforementioned algorithms as will be shown in detail later.

Keywords: Adaptive Filters, LMS Algorithm, RLS Algorithm, RI Algorithm, Correlated Noise, Impulsive Noise.

ÖZ

Uyarlanır süzgeçler, sistem parametrelerinin değişimine uyum sağlayabilmek amacıyla kullanılmaktadır. Sonlu Dürtü Yanıtlı (SDY) uyarlanır süzgeçlerde, süzgeç ağırlıkları özyineli olarak güncellenmekte ve istenilen tepki ile süzgecin çıktısındaki Ortalama Karesel Hata (OKH) en aza indirgenmeye çalışılmaktadır. Uyarlanır süzgeçlerin uygulumalarında bir çok farklı sorun bulunmaktadır: ilintili ortamlarda sabit katsayılı süzgeçler düşük başarım göstermekte, özilinti matrisinin tersinin alınmasını gerektiren algoritmalarda yakınsama ve yüksek işlem karmaşıklığı gözlemlenmekte ve zamana göre değişen ortamlarda da izleme konusunda sorunlarla karşılaşılmaktadır.

Bu tezde Özyineli Ters (ÖT), özgün bir SDY uyarlanır süzgeç önerilmektedir. Algoritma, değişken bir adım uzunluğu kullanmakta ve özilinti matrisinin anlık değerini katsayıların güncellenmesi sırasında hesaplayarak daha yüksek başarım sağlamaktadır. RLS algoritmasının aksine özilinti matrisinin tersinin hesaplanmasına ihtiyaç duyulmadığı için yakınsama daha başarılı bir şekilde dağlanmaktadır. ÖT algoritmasının ortalama öğrenme eğrisi türetilerek RLS ve LMS algoritmalarınınkilerle kıyaslanmıştır. Önerilen algoritma için hızlı bir hesaplama yöntemi de önerilmiştir. Dürtün gürültüyü gidermek amacıyla ÖT algoritmasının gürbüz sürümü de önerilmiştir. Unutma katsayısının ÖT başarımına etkisi araştırılmıştır. Ayrıca imge işleme uygulamalarında kullanılmak üzere iki boyutlu ÖT algoritması kullanılmıştır. Son olarak da ÖT algoritmasının ikinci dereceden kestirim yapan sürümü türetilmiş ve başarımı gösterilmiştir.

Başarım karşlaştırılmalarında farklı özelliklerinden dolayı LMS, NLMS, değişken adımlı LMS, DCT LMS, TDVSS, RLS, SFTRLS ve RRLS algoritmaları kullanılmıştır. Karşılaştırmalar, Gauss veya dürtün, ilintili veya beyaz gürültü ortamlarında benzetimlerle gerçekleştirilmiştir. Ayrıca, zamanda değişen gürültü ortamlarda, sistem tanımlaması ve yankı giderme uygulamalarında önerilen ÖT algoritmasının daha yüksek OKH hata başarımına ve daha düşük işlem karmaşıklığına sahip olduğu gösterilmiştir.

Anahtar Kelimeler: Uyarlanır süzgeç, LMS Algorima, RLS Algorima, dürtün gürültü, ilintili gürültü.

ACKNOWLEDGMENTS

I would like to give my sincere gratitude to my supervisors Assoc. Prof. Dr. Aykut Hocanin and Prof. Dr. Osman Kukrer for their continuous support, great guidance, endless help, knowledge and huge confidence they gave me. Many thanks to my department and fellow associates for their help and support during my course of study. My warm regards to the my wife for her presence, as it enhanced my motivation. Finally, words will not be enough to thank my family for their infinite support and patience.

Dedicated to my Family

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

β	Forgetting factor
δ	Regularization parameter
$\delta\widetilde{\mathbf{p}}$	Perturbation part of the cross-correlation vector
$\delta\widetilde{\mathbf{R}}$	Perturbation part of the autocorrelation matrix
$\delta\widetilde{\mathbf{w}}$	Perturbation part of the tap weights vector
ϵ	Impulsive noise frequency
κ	Impulsive noise strength
λ	Eigenvalue
μ	Convergence rate
σ^2	Variance
χ	Eigenvalue spread
$\mathbf{I}$	Identity matrix
N	Filter length
$\mathbf{p}$	Cross-correlation matrix
P_d	Power of the difference between the actual speech signal and the predicted one
P_e	Power of the error signal
$\mathbf{R}$	Autocorrelation matrix
$\mathbf{w}$	Tap weights vector
$\mathbf{x}$	Tap input vector
1D	1 dimensional

2D	2 dimensional
ACGN	Additive correlated Gaussian noise
AEC	Acoustic echo canceller
ALE	Adaptive line enhancer
AWGN	Additive white Gaussian noise
ACIN	Additive correlated impulsive noise
AWIN	Additive white impulsive noise
DCTLMS	Discrete cosine transform LMS
ERLE	Echo return loss enhancement
EVD	Eigenvalue decomposition
FIR	Finite impulse response
IIR	Infinite impulse response
LMS	Least-mean-square
MSE	mean-square-error
NLMS	Normalized least-mean-square
pdf	probability density function
PRRLS	Proposed robust RLS
PSF	Point spread function
RI	Recursive inverse
RLS	Recursive-least-squares
RRI	robust RI
RRLS	Robust RLS
SFTRLS	Stabilized fast transversal RLS
SNR	Signal-to-noise ratio

TDVSS	Transform domain LMS with variable step-size
VSSLMS	Variable step size LMS

Chapter 1

INTRODUCTION

1.1. Introduction

During the last decades, adaptive filtering techniques have attracted the attention of many researchers due to their properties of self-designing [1]-[4]. In applications where some a priori information about the statistics of the data is available, an optimal filter for that application can be designed in advance (i.e. the Wiener Filter that minimizes the mean-square-error (mse) between the output of the filter and a certain desired response) [1]. In the absence of this a priori information, adaptive filtering techniques possesses the ability to adapt the filter coefficients to the statistics of the signal involved. As a consequence, adaptive filters have been applied to diverse fields such as signal processing [1], communications [5], control [3], etc.

Adaptive filtering consists of two basic operations; the filtering process which generates an output signal from an input data signal, and the adaptation process; which adjusts the filter coefficients in a way to minimize a desired cost function. Basically, there is a large number of filter structures and algorithms that have been used in adaptive filtering applications.

Adaptive filters can be classified into two main categories according to their impulse response: adaptive finite impulse response (FIR) [6]; whose impulse response is of a finite duration because it settles to zero in finite time, and adaptive infinite impulse

response (IIR) filters [6]; which have internal feedback and may continue to respond indefinitely (usually decaying). IIR filters are beyond the scope of this thesis. Adaptive FIR filters have many structures, to mention a few, adaptive transversal filters, the lattice predictor, the systolic array, etc [1]. The adaptive transversal filters structure (which will be our main concern) is shown in Fig. 1.1.

The vector of tap inputs at time k is denoted by $\mathbf{x}(k)$, and the corresponding estimate of the desired response at the filter output is denoted by $\hat{d}(k|X_k)$, where X_k is the space spanned by the tap inputs $x(k), x(k-1), \ldots, x(k-N+1)$. By comparing this with the actual desired response $d(k)$, we produce an estimation error denoted by $e(k) = d(k) - \hat{d}(k|X_k)$.

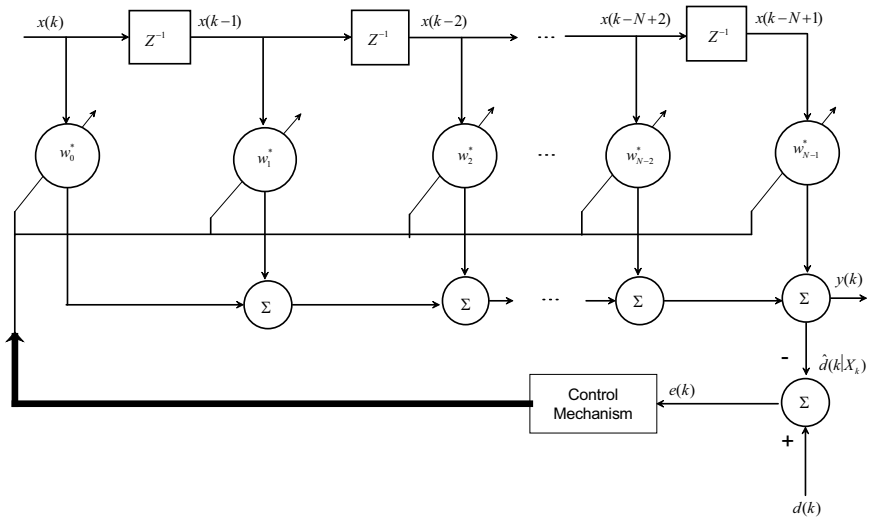

Figure 1.1. Structure of the adaptive transversal filter, [1].

1.2. Our Contributions

In this work, we propose a new FIR adaptive filtering algorithm to overcome some of the difficulties experienced with the adaptive filtering algorithms available; some of these difficulties are:

(a) Fixed-step size [7]: in highly correlated signals, the step size should be selected very small in order to follow up the changes in the environment.

(b) High computational complexity [8]: which is the number of add./sub. and mul./div. in each update iteration of the tap-weight vector.

(c) Stability [8]: adaptive algorithms that use the inverse autocorrelation matrix in it filter-tap update equation, may face stability problems because of this inverse estimate of the autocorrelation matrix.

(d) Tracking ability [9]: in non-stationary environments, adaptive algorithms are required to track the statistical variations in the environment. This would be difficult, if not impossible, in adaptive algorithms when the step-size or the forgetting factor are relatively large.

The proposed approach uses a variable step-size and the instantaneous value of the autocorrelation matrix in the coefficient update equation. This will be shown to lead to improved performance compared with the conventional approaches. Convergence analysis of the algorithm has been presented, where its shown that the proposed algorithm converges in the mean and mean square sense. The ensemble-average learning curve of the proposed algorithm is derived and compared with those of the recursive-least-squares (RLS) [1] and least-mean-square (LMS) [1], [10] algorithms. Several versions of the RI algorithm have been developed as follows:

1. A general fast implementation technique, which significantly reduces the computational complexity, for the proposed algorithm is presented.

2. A robust version of the algorithm, which leads to an improved performance in impulsive noise environments is presented.

3. A two-dimensional (2D) version of the algorithm is introduced.

4. A second-order version of the proposed approach, which provides further improvement in the performance, is proposed.

Also, the effect of the forgetting factor on the tracking ability of the algorithm is investigated.

The performance of the proposed algorithm, its fast implemented version, its robust version, its second order version and its 2D version is compared to following adaptive filtering algorithms that take place in the literature (see Chapter 2):

(a) The standard LMS

(b) The normalized LMS (NLMS) [1]

(c) The variable step size LMS (VSSLMS) [11]

(d) The discrete cosine transform LMS (DCTLMS) [12]

(e) The transform domain LMS with variable step-size (TDVSS) [13], [14]

(f) The RLS [1]

(g) The stabilized fast transversal RLS (SFTRLS) [15]

(h) The robust RLS (RRLS) [16]

(i) The proposed robust RLS [17]

in the following noise types:

(a) Additive white Gaussian noise (AWGN)

(b) Correlated Gaussian noise

(c) White impulsive noise [18]

(d) Correlated impulsive noise

in the settings of:

(a) Noise cancellation [19]

(b) System identification [20]

(c) Channel equalization [20]

(d) Echo cancellation [21]

(e) Image deconvolution [22]

in stationary and non-stationary environments [23]. Simulations show that the proposed algorithm and its variants outperform all the aforementioned algorithms as will be shown in detail in Chapter 4.

1.3. Overview of this Thesis

The contents of this thesis are organized as follows. Following the general introduction, and our contributions in Chapter 1, Chapter 2 provides a summary of some well-known adaptive filters that will be compared with the proposed techniques.

Chapter 3 introduces the proposed recursive algorithm, the full derivation of the convergence analysis of the algorithm, the derivation the ensemble-average learning curve of the algorithm, a fast implementation technique, that reduces the computational complexity, of the proposed algorithm, a new robust version of the algorithm in impulsive noise environments, the effect of the forgetting factor (β) on the tracking ability of the proposed technique, a two-dimensional (2D) version of the algorithm is introduced and, finally, a second-order (2^{nd}) version of the proposed algorithm, which provides further improvement in the performance, is derived.

In Chapter 4, the performance of the proposed techniques is compared with those of the standard LMS, NLMS, VSSLMS, DCTLMS, TDVSS, RLS, SFTRLS, RRLS and PRRLS algorithms. Simulations were performed to investigate the performances of these algorithms in AWGN, correlated Gaussian noise and white and correlated impulsive noise in:

(a) Noise cancellation

(b) System identification

(c) Channel equalization

(d) Echo cancellation

(e) Image deconvolution

in stationary and non-stationary environments. It is shown that the proposed techniques outperform all the aforementioned algorithms in the experimental settings

used. Finally, Chapter 5 summarizes the important findings of this work and concludes the thesis.

Chapter 2

ADAPTIVE FILTERS

2.1. Introduction

Adaptive filtering techniques are widely used to cope with the variations of system parameters [4]. In finite impulse response (FIR) adaptive filtering, the filter weights are updated iteratively by minimizing the MSE of the difference between the desired response of the adaptive filter and its output. Here, a summary of some well-known adaptive filtering algorithms, that will be handled during this thesis, will be provided.

2.2. Least-Mean-Square Algorithm

The well-known least-mean-square (LMS) adaptive algorithm has been used in many application areas, such as channel equalization [24], system identification [25], adaptive array processing [26], adaptive noise cancellation [4], etc. The LMS algorithm is not only simple in its filter weight updating and hardware implementation but also reasonably fast in convergence if the optimal step-size is used [1]. A summary of the algorithm is shown in Table 2.1.

2.3. Normalized Least Mean Square Algorithm

In LMS algorithm, selecting the step-size is an obstacle in many of the applications where the LMS algorithm is employed and when the input $\mathbf{x}(k)$ is large, the algorithm suffers from a gradient noise amplification problem. To overcome these problems, the normalized least-mean-square (NLMS) is proposed [27]-[30]. In NLMS,

Table 2.1 Summary of the LMS algorithm.

1. Filter Output.

   $y(k) = \hat{\mathbf{w}}^H(k)\mathbf{x}(k)$

2. Estimation Error.

   $e(k) = d(k) - y(k)$

3. Tap − Weight Adaptation.

   $\hat{\mathbf{w}}(k+1) = \hat{\mathbf{w}}(k) + \mu e^*(k)\mathbf{x}(k)$

where $\mathbf{x}(k)$ is the tap − input vector and $d(k)$ is the desired filter output.

the step-size μ is normalized by the energy of the data vector. Normalization has several interpretations [4]:

1. Corresponds to the 2^{nd}-order convergence bound.
2. Makes the algorithm independent of signal scaling.
3. Adjusts $\mathbf{w}(k+1)$ to give zero error with current input.

Table 2.2 Summary of the NLMS algorithm.

1. Filter Output.

   $y(k) = \hat{\mathbf{w}}^H(k)\mathbf{x}(k)$

2. Estimation Error.

   $e(k) = d(k) - y(k)$

3. Tap − Weight Adaptation.

   $\hat{\mathbf{w}}(k+1) = \hat{\mathbf{w}}(k) + \frac{\mu e^*(k)\mathbf{x}(k)}{\epsilon + \mathbf{x}^H(k)\mathbf{x}(k)}$

   where ϵ is a very small value to avoid dividing by zero.

So, a summary of the NLMS algorithm is given in Table 2.2.

The NLMS algorithm usually converges much more quickly than LMS at very little extra cost; NLMS is very commonly used in some applications such as acoustic echo cancelation problems [7].

The NLMS algorithm introduces a problem, that is; when the tap-input vector $\mathbf{x}(k)$ is small, numerical difficulties may arise because then we have to divide by a small value for the squared norm $\|\mathbf{x}(k)\|^2$.

2.4. Variable Step-Size LMS

A number of LMS variations have been proposed to overcome the disadvantages of a fixed step-size parameter and numerical difficulties introduced by the LMS and NLMS algorithms by employing variable step-size [31], [32] or automatic gain control schemes. The variable step size LMS (VSSLMS) algorithm [11] is one of these algorithms. A summary of the algorithm is shown in Table 2.3.

However, the LMS algorithm and its variants fail or perform poorly in some environments, especially, when the input signal is highly correlated or the additive noise is impulsive.

2.5. Transform Domain LMS Algorithms

The transform domain adaptive filtering algorithms [12], [33] - [34], [13] such as the transform domain LMS with variable step-size (TDVSS) [13] and discrete cosine transform LMS (DCTLMS) [12] are effective in correlated noise environments. These types of algorithms usually apply an orthogonal transformation on the input signal that diagonalize the autocorrelation matrix of the input signal.

Table 2.3 Summary of the VSSLMS algorithm.

1. Filter Output.

$$y(k) = \hat{\mathbf{w}}^H(k)\mathbf{x}(k)$$

2. Estimation Error.

$$e(k) = d(k) - y(k)$$

3. Tap − Weight Adaptation.

$$\hat{\mathbf{w}}(k+1) = \hat{\mathbf{w}}(k) + \mu(k)e^*(k)\mathbf{x}(k)$$

$$\text{where } \mu(k+1) = \begin{cases} \mu_{max}, & \text{if } \alpha\mu(k) + \gamma e^2(k) > \mu_{max} \\ \mu_{min}, & \text{if } \alpha\mu(k) + \gamma e^2(k) < \mu_{min} \\ \alpha\mu(k) + \gamma e^2(k), & \text{otherwise} \end{cases}$$

$$\text{where } 0 < \alpha < 1;\ \ \gamma > 0;\ \ 0 < \mu_{min} < \mu_{max};\ \text{ and } \mu_{max} = \mu_0.$$

2.5.1. Discrete Cosine Transform LMS

In DCTLMS algorithm [12], DCT is firstly applied to the tap-input vector $\mathbf{x}(k)$. This transformation leads to an enhanced performance (as will be shown in Chapter 4) in terms of MSE compared to the algorithms mentioned in Sections 2.2, 2.3 and 2.4. A summary of the algorithm is shown in Table 2.4.

Although the DCTLMS algorithm provides better performance than the algorithms in Sections 2.2, 2.3 and 2.4, it suffers from its high computational complexity, due to the transformation process, and slow convergence rate.

2.5.2. Transform Domain LMS with Variable Step-Size

With a comparable computational complexity to that of the DCTLMS algorithm the TDVSS algorithm is recently proposed by Bilc, Kuosmanen and Egiazarian [13] to provide much faster convergence rate. The algorithm uses the output error to update

Table 2.4 Summary of the DCTLMS algorithm.

1. Discrete Cosine Transform.

   $u_k(i) = \sum_{l=0}^{N-1} \sqrt{\frac{2}{N}} K_i cos\left(\frac{i(l+1/2)\pi}{N}\right) x(k-l)$; where $i = 0, \ldots, N-1$

   with $K_i = \frac{1}{\sqrt{2}}$ for $i = 0$ and 1 otherwise.

2. Power Normalization.

   $v_k(i) = \frac{u_k(i)}{\sqrt{P_k(i)+\epsilon}}$; where $i = 0, \ldots, N-1$

   where ϵ is a small constant.

   $P_k(i) = \gamma P_{k-1}(i) + (1-\gamma)\, u_k^2(i)$; where $i = 0, \ldots, N-1$

   where $\gamma \in]0, 1]$ is generally chosen close to 1.

3. LMS Filtering.

   $\hat{\mathbf{w}}(k+1) = \hat{\mathbf{w}}(k) + \mu e(k)\mathbf{v}^*(k)$

   where $e(k) = d(k) - \hat{\mathbf{w}}^H(k)\mathbf{v}(k)$.

the step-size of the transform domain LMS algorithms. A summary of the algorithm is shown in Table 2.5.

Even though the TDVSS algorithm outperforms the DCTLMS, also it suffers from its high computational complexity due to the transformation process, and has many parameters to be carefully selected.

2.6. Recursive-Least-Squares Algorithm

The recursive-least-squares (RLS) algorithm [1], [35]-[36] was proposed to offer superior performance compared to that of the LMS algorithm and its variants [37]-[42], with few parameters to be predefined compared to the DCTLMS and TDVSS algorithms; especially in highly correlated environments. In the RLS algorithm, an estimate of the autocorrelation matrix is used to decorrelate the current input

data. Also, the quality of the steady-state solution keeps on improving over time, eventually leading to an optimal solution. A summary of the algorithm is shown in Table 2.6.

Although the RLS algorithm provides good performance in such environments, it suffers from relatively high computational complexity $O(N^2)$. In the RLS algorithm, the forgetting factor (β) has to be chosen such that its value is very close to unity to ensure the stability and convergence of the algorithm. However, this poses a limitation for the use of the algorithm since small values of β may be required for signal tracking in non-stationary environments [8], [43].

2.7. Stabilized Fast Transversal RLS Algorithm

The stabilized fast transversal RLS (SFTRLS) [37], [38], [44]-[45] algorithm has been proposed to reduce the computational complexity of the RLS algorithm to an order of $O(N)$ [44]. A summary of the algorithm is shown in Table 2.7.

The disadvantage of the SFTRLS algorithm is its high MSE even if the signal-to-noise ratio (SNR) is relatively low and requires many parameters to be defined. In addition to that, the RLS algorithm and its variants may face numerical stability problems due to the update of the inverse autocorrelation matrix [8], [46].

2.8. Robust RLS Algorithm

The robust RLS (RRLS) [16] algorithm was proposed to provide better performance than the RLS algorithm. The algorithm uses a variable forgetting factor that leads to a faster tracking ability. A summary of the algorithm is shown in Table 2.8.

Even though the RRLS algorithm provides better performance than the aforementioned algorithms, it suffers from its very high computational complexity ($O(N^3)$).

Also, the performance of all the previously mentioned algorithms is poor in impulsive noise environments.

2.9. Proposed Robust RLS Algorithm

Several robust versions of the RLS algorithm for impulsive noise environments have been proposed, [17], [47]-[49]. One example is the recently proposed robust RLS algorithm [17]. A summary of the algorithm is shown in Table 2.9.

All of the RLS algorithm and its variants suffer from their high computational complexity, and stability problems due to the estimate of the inverse of the autocorrelation matrix [8].

Due to all of the aforementioned reasons, a new recursive inverse (RI) algorithm has been introduced [50]-[53] to overcome some of the difficulties experienced.

Table 2.5 Summary of the TDVSS algorithm.

1. Discrete Cosine Transform.

$u_k(i) = \sum_{l=0}^{N-1} \sqrt{\frac{2}{N}} K_i cos\left(\frac{i(l+1/2)\pi}{N}\right) x(k-l)$; where $i = 0, \ldots, N-1$

with $K_i = \frac{1}{\sqrt{2}}$ for $i = 0$ and 1 otherwise.

2. Filtering Process.

$\hat{w}_i(k+1) = \hat{w}_i(k) + \frac{\mu(k)}{\epsilon + \sigma_i^2(k)} e(k) u_i^*(k)$

where

$e(k) = d(k) - \hat{\mathbf{w}}^H(k)\mathbf{u}(k)$,

$\sigma_i^2(k)$ is the power estimate of the i^{th} transform coefficient $u_i(k)$,

$\hat{w}_i(k)$ is the i^{th} coefficient of the adaptive filter,

ϵ is a small constant that eliminates the overflow when the values of $\sigma_i^2(k)$ are small,

$\sigma_i^2(k) = \beta\sigma_i^2(k-1) + (1-\beta)|u_i(k)|^2$, where $\beta \in [0,1]$ is the forgetting factor

$$\mu(k+1) = \begin{cases} A(k), & \text{if } k = nL, \text{ and } A(k) \in (\mu_{max}, \mu_{min}) \\ \mu_{max}, & \text{if } k = nL, \text{ and } A(k) \geq \mu_{max} \\ \mu_{min}, & \text{if } k = nL, \text{ and } A(k) \leq \mu_{min}) \\ \mu(k), & \text{if } k \neq nL \end{cases}$$

where $A(k) = \alpha\mu(k) + \frac{\gamma}{L} \sum_{i=k-L+1}^{k} e^2(i)$, where $k = 1, 2, \ldots$

$\alpha,\ \gamma,\ L,\ \mu_{min},\ \mu_{max}$ are predifined parameters.

Table 2.6 Summary of the RLS algorithm.

Initialize the algorithm by setting,

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

$$\mathbf{P}(0) = \delta^{-1}\mathbf{I}$$

and

$$\delta = \begin{cases} \text{small positive constant for high SNR} \\ \text{large positive constant for low SNR.} \end{cases}$$

for each instant of time, $k = 1, 2, \ldots$ compute

$$\mathbf{S}(k) = \mathbf{P}(k-1)\mathbf{x}(k)$$

$$\mathbf{k}(k) = \frac{\mathbf{S}(k)}{\beta + \mathbf{x}^H(k)\mathbf{S}(k)}$$

$$\xi(k) = d(k) - \hat{\mathbf{w}}^H(k-1)\mathbf{x}(k)$$

$$\hat{\mathbf{w}}(k) = \hat{\mathbf{w}}(k-1) + \mathbf{k}(k)\xi^*(k)$$

and

$$\mathbf{P}(k) = \beta^{-1}\mathbf{P}(k-1) - \beta^{-1}\mathbf{k}(k)\mathbf{x}^H(k)\mathbf{P}(k-1).$$

Table 2.7 Summary of the SFTRLS algorithm

1. Initialization

$\mathbf{w}_f(-1,N) = \mathbf{w}_b(-1,N) = \mathbf{w}(-1,N) = \mathbf{0}$

$\hat{\phi}(-1,N) = \mathbf{0},\ \gamma(-1,N,3) = 1$

$\xi^d_{b_{min}}(-1,N) = \xi^d_{f_{min}}(-1,N) = \epsilon$ a small positive constant

$\kappa_1 = 1.5,\ \kappa_2 = 2.5,\ \kappa_3 = 1.$

2. Prediction Part

for each instant of time, $k = 1, 2, \ldots$

$$e_f(k,N) = \mathbf{x}^T(k,N+1)\begin{bmatrix} 1 \\ -\mathbf{w}_f(k-1,N) \end{bmatrix}$$

$$\varepsilon_f(k,N) = e_f(k,N)\gamma(k-1,N,3)$$

$$\hat{\phi}(k,N+1) = \begin{bmatrix} 0 \\ \hat{\phi}(k-1,N) \end{bmatrix} + \frac{1}{\beta\xi^d_{f_{min}}(k-1,N)}\begin{bmatrix} 1 \\ -\mathbf{w}_f(k-1,N) \end{bmatrix} e_f(k,N)$$

$$\gamma^{-1}(k,N+1,1) = \gamma^{-1}(k-1,N,3) + \phi_0(k,N+1)e_f(k,N)$$

$$\left[\xi^d_{f_{min}}(k,N)\right]^{-1} = \beta^{-1}\left[\xi^d_{f_{min}}(k-1,N)\right]^{-1} - \gamma(k,N+1,1)\hat{\phi}^2_0(k,N+1)$$

$$\mathbf{w}_f(k,N) = \mathbf{w}_f(k-1,N) + \hat{\phi}(k-1,N)\varepsilon(k,N)$$

$$e_b(k,N,1) = \beta\xi^d_{b_{min}}(k-1,N)\hat{\phi}_{N+1}(k,N+1)$$

$$e_b(k,N,2) = \left[\mathbf{w}^T_b(k-1,N),\ 1\right]\mathbf{x}(k,N+1)$$

$$e_{b,i}(k,N,3) = e_b(k,N,2)\kappa_i + e_b(k,N,1)\left[1-\kappa_i\right] \text{ for } i = 1,2,3$$

$$\gamma^{-1}(k,N,2) = \gamma^{-1}(k,N+1,1) - \hat{\phi}_{N+1}(k,N+1)e_{b,3}(k,N,3)$$

$$\varepsilon b,j(k,N,3) = e_{b,j}(k,N,3)\gamma(k,N,2) \text{ for } j = 1,2$$

$$\xi^d_{b_{min}}(k,N) = \beta\xi^d_{b_{min}}(k-1,N) + \xi_{b,2}(k,N,3)e_{b,2}(k,N,3)$$

$$\begin{bmatrix} \hat{\phi}(k,N) \\ 0 \end{bmatrix} = \hat{\phi}(k,N+1) - \hat{\phi}_{N+1}(k,N+1)\begin{bmatrix} -\mathbf{w}_b(k-1,N) \\ 1 \end{bmatrix}$$

Table 2.7 Summary of the SFTRLS algorithm (Continued).

$\mathbf{w}_b(k,N) = \mathbf{w}_b(k-1,N) + \hat{\phi}(k,N)\varepsilon_{b,1}(k,N,3)$

$\gamma^{-1}(k,N,3) = 1 + \hat{\phi}^T(k,N)\mathbf{x}(k,N)$

3. Joint − Process Estimation

$e(k,N) = d(k) - \mathbf{w}^T(k-1,N)\mathbf{x}(k,N)$

$\varepsilon(k,N) = e(k,N)\gamma(k,N,3)$

$\mathbf{w}(k,N) = \mathbf{w}(k-1,N) + \hat{\phi}(k,N)\varepsilon(k,N)$

Table 2.8 Summary of the RRLS algorithm.

Initialize the algorithm by setting,

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

$\mathbf{P}(0) = c\mathbf{I},\ c > 1$

for each instant of time, $k = 1, 2, \ldots$ compute

$\mathbf{P}(k+1) = \mathbf{P}(k) + \frac{\mathbf{P}(k)\mathbf{P}(k)}{\gamma^2} - \frac{\mathbf{P}(k)\mathbf{x}(k+1)\mathbf{x}^T(k+1)\mathbf{P}(k)}{1+\mathbf{x}^T(k+1)\mathbf{P}(k)\mathbf{x}(k+1)}$

$e(k+1) = d(k+1) - \mathbf{x}^T(k+1)\hat{\mathbf{w}}(k)$

$\hat{\mathbf{w}}(k+1) = \left[\mathbf{I} + \frac{\mathbf{P}(k)}{\gamma^2}\right]\hat{\mathbf{w}}(k) + \mathbf{P}(k+1)\mathbf{x}(k+1)e(k)$

where $\gamma > 0$.

Table 2.9 Summary of the PRRLS algorithm.

for each instant of time, $k = 1, 2, \ldots$ compute

$\mathbf{P}(k) = \frac{1}{\beta}\left[\mathbf{P}(k-1) + \frac{\mathbf{P}(k-1)\mathbf{x}(k)\mathbf{x}^T(k)\mathbf{P}(k-1)}{\frac{\beta}{\delta_k}+\mathbf{x}^T(k)\mathbf{P}(k-1)\mathbf{x}(k)}\right]$

$\hat{\mathbf{w}}(k) = \hat{\mathbf{w}}(k-1) + \frac{\mathbf{P}(k-1)\mathbf{x}(k)e(k)}{\frac{\beta}{\delta_k}+\mathbf{x}^T(k)\mathbf{P}(k-1)\mathbf{x}(k)}$

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

where $\delta_k = min\left(1, \frac{1}{\|\mathbf{x}(k)e(k)\|_1}\right)$.

Chapter 3

THE RECURSIVE INVERSE ADAPTIVE FILTERING ALGORITHM

3.1. Introduction

As mentioned before, the well-known RLS algorithm offers a superior speed of convergence compared to the LMS algorithm and its variants [54], [9], [39], especially in highly correlated and nonstationary environments. On the other hand the RLS algorithm suffers from its high computational complexity and tracking ability when the filter length (N) and the forgetting factor (β) [55], [56] are high. Hence, we propose a new recursive algorithm with performance comparable to the RLS algorithm, but with much lower complexity.

This chapter will be organized as follows. In section 3.2, the new recursive algorithm is derived. In Section 3.3, the convergence analysis of the algorithm is presented. In section 3.4, a comparison between the LMS and the recursive inverse (RI) algorithms is made. In section 3.5, the ensemble-average learning curve of the RI algorithm is derived. In section 3.6, a fast implementation technique, which considerably decreases the computational burden of the RI algorithm, is introduced. In section 3.7, a robust version of the RI algorithm in impulsive noise environments is presented. In section 3.8, the effect of the forgetting factor (β) on the performance of the RI algorithm is investigated. In section 3.9, a two-dimensional (2D) version of the RI algorithm, for image processing applications, is introduced. Finally,

in section 3.10, a second-order (2^{nd}) version of the RI algorithm, which provides further improvement in the performance, is derived.

3.2. The Recursive Inverse (RI) Algorithm

In this section, the recursive update equations for the correlation matrices [37] which are used in the Newton-LMS [57]-[61] and the RLS [54]-[56] algorithms will first be presented. The derivation of the RI algorithm will then follow. It is known that the Wiener-Hopf equation [1] leads to the optimum solution for the FIR filter coefficients. The coefficients are given by

$$\mathbf{w}(k) = \mathbf{R}^{-1}(k)\mathbf{p}(k), \tag{3.1}$$

where k is the time parameter ($k = 1, 2, \ldots$), $\mathbf{w}(k)$ is the filter weight vector calculated at time k, $\mathbf{R}(k)$ is the estimate of the tap-input vector autocorrelation matrix, and $\mathbf{p}(k)$ is the estimate of the cross-correlation vector between the desired output signal and the tap-input vector. The solution of (3.1) is required at each iteration where the filter coefficients are updated. As an additional requirement, the autocorrelation matrix should be nonsingular at each iteration, [1].

Reconsidering (3.1) where the correlations are estimated recursively [37] as;

$$\mathbf{R}(k) = \beta\mathbf{R}(k-1) + \mathbf{x}(k)\mathbf{x}^T(k), \tag{3.2}$$

and

$$\mathbf{p}(k) = \beta\mathbf{p}(k-1) + d(k)\mathbf{x}(k), \tag{3.3}$$

where β is the forgetting factor which is usually very close to one.

The estimate in (3.2) would be definitely singular for $k < N$, (N is the filter length), and therefore, equation (3.1) can not be applied. The estimate becomes nonsingular for $k \geq N$. Under this condition substituting (3.2) and (3.3) in (3.1) yields,

$$\mathbf{w}(k) = \{\beta\mathbf{R}(k-1) + \mathbf{x}(k)\mathbf{x}^T(k)\}^{-1}[\beta\mathbf{p}(k-1) + d(k)\mathbf{x}(k)], \tag{3.4}$$

by using the matrix inversion lemma [62] (see appendix), equation (3.4) becomes

$$\begin{aligned}
\mathbf{w}(k) &= \frac{1}{\beta}\left[\mathbf{R}^{-1}(k-1) - \frac{\mathbf{R}^{-1}(k-1)\mathbf{x}(k)\mathbf{x}^T(k)\mathbf{R}^{-1}(k-1)}{\beta + \mathbf{x}^T(k)\mathbf{R}^{-1}(k-1)\mathbf{x}(k)}\right] \\
&\times \quad (\beta\mathbf{p}(k-1) + d(k)\mathbf{x}(k)) \\
&= \left[\mathbf{I} - \frac{\mathbf{R}^{-1}(k-1)\mathbf{x}(k)\mathbf{x}^T(k)}{\beta + \mathbf{x}^T(k)\mathbf{R}^{-1}(k-1)\mathbf{x}(k)}\right]\mathbf{w}(k-1) \\
&+ \frac{1}{\beta}\left[\frac{\mathbf{I}\{\beta + \mathbf{x}^T(k)\mathbf{R}^{-1}(k-1)\mathbf{x}(k)\}\mathbf{R}^{-1}(k-1)\mathbf{x}(k)}{\beta + \mathbf{x}^T(k)\mathbf{R}^{-1}(k-1)\mathbf{x}(k)}\right] d(k) \\
&+ \frac{1}{\beta}\left[\frac{\mathbf{R}^{-1}(k-1)\mathbf{x}(k)\mathbf{x}^T(k)\mathbf{R}^{-1}(k-1)\mathbf{x}(k)}{\beta + \mathbf{x}^T(k)\mathbf{R}^{-1}(k-1)\mathbf{x}(k)}\right] d(k).
\end{aligned} \tag{3.5}$$

Rearranging (3.5);

$$\begin{aligned}
\mathbf{w}(k) &= \left[\mathbf{I} - \frac{\mathbf{R}^{-1}(k-1)\mathbf{x}(k)\mathbf{x}^T(k)}{\beta + \mathbf{x}^T(k)\mathbf{R}^{-1}(k-1)\mathbf{x}(k)}\right]\mathbf{w}(k-1) \\
&+ \left[\frac{\mathbf{R}^{-1}(k-1)\mathbf{x}(k)}{\beta + \mathbf{x}^T(k)\mathbf{R}^{-1}(k-1)\mathbf{x}(k)}\right] d(k) \\
&= \mathbf{w}(k-1) + \mu(k)\mathbf{R}^{-1}(k-1)\mathbf{x}(k)e(k)
\end{aligned} \tag{3.6}$$

The update equation in (3.6) is the Newton-LMS [57]-[61] algorithm where the

a-priori filtering error is,

$$e(k) = d(k) - \mathbf{x}^T(k)\mathbf{w}(k-1), \tag{3.7}$$

and the variable step-size is

$$\mu(k) = \frac{1}{\beta + \mathbf{x}^T(k)\mathbf{R}^{-1}(k-1)\mathbf{x}(k)}.$$

Newton-LMS [57]-[61] is equivalent to the Wiener solution with exponential-forgetting window estimation of the autocorrelation and cross-correlation. However, Newton-LMS requires the inverse of the autocorrelation matrix. In the RI algorithm, which will be derived, this is avoided.

The RLS algorithm is similar to Newton-LMS, except that the updating of the correlations are not performed directly using (3.2) and (3.3). Instead, the inverse autocorrelation matrix is updated to avoid inverting it at each step [1]. The basic idea behind the RI algorithm is the iterative solution of the Wiener equation (3.1) at each time step k. Specifically, the following iteration converges to the Wiener Solution,

$$\mathbf{w}_{n+1}(k) = [\mathbf{I} - \mu\mathbf{R}(k)]\mathbf{w}_n(k) + \mu\mathbf{p}(k), \quad n = 0, 1, 2, \ldots \tag{3.8}$$

if μ satisfies the convergence criterion [1];

$$\mu < \frac{2}{\lambda_{max}(\mathbf{R}(k))}. \tag{3.9}$$

Considering the update equations for the correlations, and taking the expectation of

$\mathbf{R}(k)$ in (3.2) we get;

$$\bar{\mathbf{R}}(k+1) = \beta\bar{\mathbf{R}}(k) + \mathbf{R}_{xx}, \tag{3.10}$$

where $\mathbf{R}_{xx} = E\left\{\mathbf{x}(k)\mathbf{x}^T(k)\right\}$ and $\bar{\mathbf{R}}(k) = E\{\mathbf{R}(k)\}$. Solving (3.10) yields,

$$\bar{\mathbf{R}}(k) = \frac{1-\beta^k}{1-\beta}\mathbf{R}_{xx}, \tag{3.11}$$

and as $k \rightarrow \infty$ (see appendix)

$$\bar{\mathbf{R}}(\infty) = \frac{1}{1-\beta}\mathbf{R}_{xx}. \tag{3.12}$$

Equation (3.11) implies that the eigenvalues of the estimated autocorrelation matrix increase exponentially, and in the limit become $\frac{1}{1-\beta}$ times that of the original matrix. The implication is that since μ must be chosen to satisfy (3.9) in the limit as well, we get:

$$\mu < \frac{2(1-\beta)}{\lambda_{max}(\mathbf{R}_{xx})}, \tag{3.13}$$

equation (3.13) restricts μ to values much smaller than the one required in (3.8) if $\mathbf{R}_{xx}$ was used instead of $\mathbf{R}(k)$. Hence, it would be advantageous to make the step-size μ variable so that,

$$\mu(k) < \frac{2}{\lambda_{max}(\mathbf{R}(k))} \quad = \quad \left(\frac{1-\beta}{1-\beta^k}\right)\left(\frac{2}{\lambda_{max}(\mathbf{R}_{xx})}\right) = \frac{\mu_{max}}{1-\beta^k}, \tag{3.14}$$

or equivalently,

$$\mu(k) = \frac{\mu_0}{1 - \beta^k} \qquad \text{where } \mu_0 < \mu_{max}. \tag{3.15}$$

The iteration in (3.8) has a high computational cost. Therefore, with the variable step-size, only one iteration at each time step may be sufficient. Finally, the weight update equation for the proposed RI algorithm becomes:

$$\mathbf{w}(k) = [\mathbf{I} - \mu(k)\mathbf{R}(k)]\mathbf{w}(k-1) + \mu(k)\mathbf{p}(k). \tag{3.16}$$

The RI algorithm has a major advantage compared to the RLS algorithm in that it does not require the updating of the inverse autocorrelation matrix. Also, its computational complexity is much less than that of the RLS as will be shown later. Due to the update of the inverse autocorrelation matrix, RLS type algorithms may face numerical stability problems [37], which is not the case for the RI algorithm.

The essence of the RI algorithm is the recursive inversion of the autocorrelation matrix, simultaneously, with its recursive estimation. For the convergence of the filter weight vector $\mathbf{w}(k)$, the correlations in (3.2) and (3.3) must converge, and this convergence is a slow process since beta is close to unity. This means that the iterative inversion of $\mathbf{R}(k)$ in (3.8) need not be carried until convergence (for large n), since the iteration will be continued in the next time step with slight changes in the correlation estimates. It should be noted here that, if the iteration in (3.8) is continued till convergence, then the RI algorithm's convergence rate becomes the same as that of RLS. The main difference between the two algorithms is in how the

inverse of the autocorrelation matrix estimate at time step k is calculated. In the RLS, this is obtained from a recursion in terms of the inverse, and in the RI, it is obtained from an iterative scheme (as in (3.8)) which does not involve the inverse. It is expected that, with the updating of the correlations in (3.2) and (3.3) being sufficiently slow, the iterative inversion of the autocorrelation matrix will approach the true inverse in a small number of time steps, and will track it closely afterwards.

3.3. Convergence Analysis of the RI Algorithm

By solving (3.2) we get;

$$\mathbf{R}_k = \sum_{i=0}^{k} \beta^{k-i} \mathbf{x}(i)\mathbf{x}^T(i), \tag{3.17}$$

Substituting (3.17) in (3.16);

$$\mathbf{w}_k = \left[\mathbf{I} - \mu_k \sum_{i=0}^{k} \beta^{k-i} \mathbf{x}(i)\mathbf{x}^T(i)\right] \mathbf{w}_{k-1} + \mu_k \mathbf{p}_k, \tag{3.18}$$

where

$$\mathbf{x}(i)\mathbf{x}^T(i) = \mathbf{R}_{xx} + \delta\mathbf{R}(i), \tag{3.19}$$

where $\delta\mathbf{R}(i)$ is the random part of the autocorrelation matrix. Substituting (3.19) in (3.17) gives,

$$\begin{aligned} \mathbf{R}_k &= \sum_{i=0}^{k} \beta^{k-i} \mathbf{R}_{xx} + \sum_{i=0}^{k} \beta^{k-i} \delta\mathbf{R}(i) \\ &= \left(\frac{1-\beta^{k+1}}{1-\beta}\right) \mathbf{R}_{xx} + \sum_{i=0}^{k} \beta^{k-i} \delta\mathbf{R}(i), \end{aligned} \tag{3.20}$$

Letting,

$$\mu_k = \left(\frac{\mu_0(1-\beta)}{1-\beta^{k+1}} \right),$$

and multiplying both sides of (3.20) by μ_k yields,

$$\mu_k \mathbf{R}_k = \mu_0 \mathbf{R}_{xx} + \left(\frac{\mu_0(1-\beta)}{1-\beta^{k+1}} \right) \sum_{i=0}^{k} \beta^{k-i} \delta\mathbf{R}(i), \tag{3.21}$$

Defining,

$$\delta\widetilde{\mathbf{A}}_k = \left(\frac{\mu_0(1-\beta)}{1-\beta^{k+1}} \right) \sum_{i=0}^{k} \beta^{k-i} \delta\mathbf{R}(i)$$

and $\mathbf{A}_k \equiv \mu_k \mathbf{R}_k$,

$$\mathbf{A}_k = \mu_0 \mathbf{R}_{xx} + \delta\widetilde{\mathbf{A}}_k. \tag{3.22}$$

Following the same procedure for the cross-correlation vector, results in

$$\mu_k \mathbf{p}_k = \mu_0 \bar{\mathbf{p}} + \delta\widetilde{\mathbf{p}}_k. \tag{3.23}$$

Now, substituting (3.22) and (3.23) in (3.18) yields,

$$\mathbf{w}_k = \left[\mathbf{I} - \left(\mu_0 \mathbf{R}_{xx} + \delta\widetilde{\mathbf{A}}_k \right) \right] \mathbf{w}_{k-1} + \mu_0 \bar{\mathbf{p}} + \delta\widetilde{\mathbf{p}}_k. \tag{3.24}$$

Let

$$\mathbf{w}_k = \bar{\mathbf{w}}_k + \delta\widetilde{\mathbf{w}}_k, \tag{3.25}$$

where $\bar{\mathbf{w}}_k = E\{\mathbf{w}_k\}$ and $\delta\widetilde{\mathbf{w}}_k$ is the stochastic part of $\mathbf{w}_k$. By substituting (3.25) in (3.24),

$$\bar{\mathbf{w}}_k + \delta\widetilde{\mathbf{w}}_k = \left[\mathbf{I} - \mu_0\mathbf{R}_{xx}\right]\left(\bar{\mathbf{w}}_{k-1} + \delta\widetilde{\mathbf{w}}_{k-1}\right) - \delta\widetilde{\mathbf{A}}_k\left(\bar{\mathbf{w}}_{k-1} + \delta\widetilde{\mathbf{w}}_{k-1}\right) + \mu_0\bar{\mathbf{p}} + \delta\widetilde{\mathbf{p}}_k, \tag{3.26}$$

Subdividing (3.26) into deterministic and stochastic components,

$$\bar{\mathbf{w}}_k = \left[\mathbf{I} - \mu_0\mathbf{R}_{xx}\right]\bar{\mathbf{w}}_{k-1} + \mu_0\bar{\mathbf{p}}, \tag{3.27}$$

and

$$\delta\widetilde{\mathbf{w}}_k = \left[\mathbf{I} - \left(\mu_0\mathbf{R}_{xx} + \delta\widetilde{\mathbf{A}}_k\right)\right]\delta\widetilde{\mathbf{w}}_{k-1} + \delta\widetilde{\mathbf{p}}_k - \delta\widetilde{\mathbf{A}}_k\bar{\mathbf{w}}_{k-1}, \tag{3.28}$$

are obtained. Now, let

$$\bar{\mathbf{w}}_k = \mathbf{w}_0 + \Delta\mathbf{w}_k, \tag{3.29}$$

where $\mathbf{w}_0$ is the optimum solution of $\mathbf{w}_k$. Keeping in mind that,

$$\bar{\mathbf{p}} = \mathbf{R}_{xx}\mathbf{w}_0, \tag{3.30}$$

substituting (3.29) and (3.30) in (3.27),

$$\begin{aligned}\mathbf{w}_0 + \Delta\mathbf{w}_k &= [\mathbf{I} - \mu_0\mathbf{R}_{xx}]\,(\mathbf{w}_0 + \Delta\mathbf{w}_{k-1}) + \mu_0\bar{\mathbf{p}} \\ &= \mathbf{w}_0 - \mu_0\mathbf{R}_{xx}\mathbf{w}_0 + [\mathbf{I} - \mu_0\mathbf{R}_{xx}]\,\Delta\mathbf{w}_{k-1} + \mu_0\mathbf{R}_{xx}\mathbf{w}_0. \quad (3.31)\end{aligned}$$

Rearranging (3.31),

$$\Delta\mathbf{w}_k = [\mathbf{I} - \mu_0\mathbf{R}_{xx}]\,\Delta\mathbf{w}_{k-1} \quad (3.32)$$

is obtained. Equation (3.32) is a linear time-invariant equation and its solution is,

$$\Delta\mathbf{w}_k = [\mathbf{I} - \mu_0\mathbf{R}_{xx}]^k\,\Delta\mathbf{w}(0). \quad (3.33)$$

From (3.33) we notice that $\Delta\mathbf{w}_k \rightarrow \mathbf{0}$ as $k \rightarrow \infty$ if $|\lambda\,(\mathbf{I} - \mu_0\mathbf{R}_{xx})\,| < 1$, which shows that the coefficients converge to their optimum solution in the mean sense.

Now, by rearranging (3.28) we can write,

$$\delta\widetilde{\mathbf{w}}_k = [\mathbf{I} - \mu_0\mathbf{R}_{xx}]\,\delta\widetilde{\mathbf{w}}_{k-1} - \delta\widetilde{\mathbf{A}}_k\mathbf{w}_{k-1} + \delta\widetilde{\mathbf{p}}_k, \quad (3.34)$$

Let us consider the last two terms of the right hand side of (3.34),

$$-\delta\widetilde{\mathbf{A}}_k\mathbf{w}_{k-1} + \delta\widetilde{\mathbf{p}}_k = -\mu_k\sum_{i=0}^{k}\beta^{k-i}\delta\mathbf{R}(i)\mathbf{w}_{k-1} + \mu_k\sum_{i=0}^{k}\beta^{k-i}\delta\mathbf{p}(i), \quad (3.35)$$

Substituting (3.20), (3.23), (3.25) and (3.29) in (3.35) gives,

$$
\begin{aligned}
-\delta\widetilde{\mathbf{A}}_k\mathbf{w}_{k-1} + \delta\widetilde{\mathbf{p}}_k &= -\mu_k \sum_{i=0}^{k} \beta^{k-i} \left(\mathbf{x}(i)\mathbf{x}^T(i) - \mathbf{R}_{xx}\right)\left(\mathbf{w}_0 + \Delta\mathbf{w}_{k-1} + \delta\widetilde{\mathbf{w}}_{k-1}\right) \\
&+ \mu_k \left[\sum_{i=0}^{k} \beta^{k-i} \left(\mathbf{x}(i)d(i) - \bar{\mathbf{p}}\right)\right] \\
&= \mu_k \sum_{i=0}^{k} \beta^{k-i}\mathbf{x}(i)\left(d(i) - \mathbf{x}^T(i)\mathbf{w}_0\right) \\
&- \mu_k \sum_{i=0}^{k} \beta^{k-i}\bar{\mathbf{p}} + \mu_k \sum_{i=0}^{k} \beta^{k-i}\mathbf{R}_{xx}\mathbf{w}_0 \\
&+ \mu_k \sum_{i=0}^{k} \beta^{k-i}\mathbf{R}_{xx}\left(\Delta\mathbf{w}_{k-1} + \delta\widetilde{\mathbf{w}}_{k-1}\right) \\
&- \mu_k \sum_{i=0}^{k} \beta^{k-i}\mathbf{x}(i)\mathbf{x}^T(i)\left(\Delta\mathbf{w}_{k-1} + \delta\widetilde{\mathbf{w}}_{k-1}\right) \\
&= \mu_k \sum_{i=0}^{k} \beta^{k-i}\mathbf{x}(i)e_0(i) \\
&- \mu_k \sum_{i=0}^{k} \beta^{k-i}\delta\mathbf{R}(i)\left(\Delta\mathbf{w}_{k-1} + \delta\widetilde{\mathbf{w}}_{k-1}\right). \qquad (3.36)
\end{aligned}
$$

The second term in the right-hand-side of equation (3.36), is weighted time averaging and by the ergodicity assumption is equal to its expected value. Furthermore, this term can be neglected based on the independence assumption used in the analysis of the LMS algorithm [1]. Now, defining the error to be;

$$e_0 = d(i) - \mathbf{x}^T(i)\mathbf{w}_0,$$

and by the orthogonality property between $\mathbf{x}(k)$ and $e_0(k)$, the result of the first term in equation (3.36), which is a weighted time averaging and is equal to the expectation for an ergodic process, becomes zero. (The necessary and sufficient condition for the cost function to attain its minimum value is for the corresponding value of

the estimation error $e_0(k)$ to be orthogonal to each input sample that enters into the estimation of the desired response at time k), [1]. Therefore, (3.34) becomes,

$$\delta\widetilde{\mathbf{w}}_k = [\mathbf{I} - \mu_0\mathbf{R}_{xx}]\,\delta\widetilde{\mathbf{w}}_{k-1}, \tag{3.37}$$

In order to find the covariance matrix of $\delta\widetilde{\mathbf{w}}_k$,

$$E\left\{\delta\widetilde{\mathbf{w}}_k\delta\widetilde{\mathbf{w}}_k^T\right\} = [\mathbf{I} - \mu_0\mathbf{R}_{xx}]\,E\left\{\delta\widetilde{\mathbf{w}}_{k-1}\delta\widetilde{\mathbf{w}}_{k-1}^T\right\}[\mathbf{I} - \mu_0\mathbf{R}_{xx}] \tag{3.38}$$

is obtained. Defining $\mathbf{Q}_k = E\left\{\delta\widetilde{\mathbf{w}}_k\delta\widetilde{\mathbf{w}}_k^T\right\}$,

$$\mathbf{Q}_k = [\mathbf{I} - \mu_0\mathbf{R}_{xx}]\,\mathbf{Q}_{k-1}\,[\mathbf{I} - \mu_0\mathbf{R}_{xx}]\,. \tag{3.39}$$

Solving (3.39) yields,

$$\mathbf{Q}_k = [\mathbf{I} - \mu_0\mathbf{R}_{xx}]^k\,\mathbf{Q}_0\,[\mathbf{I} - \mu_0\mathbf{R}_{xx}]^k\,. \tag{3.40}$$

It can be observed that $\mathbf{Q}_k \rightarrow \mathbf{0}$ as $k \rightarrow \infty$ if $|\lambda\,(\mathbf{I} - \mu_0\mathbf{R}_{xx})\,| < 1$. Along with the previous result of $\Delta\mathbf{w}_k \rightarrow \mathbf{0}$ as $k \rightarrow \infty$, if $|\lambda\,(\mathbf{I} - \mu_0\mathbf{R}_{xx})\,| < 1$ this assures convergence in the mean square sense. It should be noted here that the result in (3.40), which implies that the steady state covariance of the weight vector error is zero, is the consequence of the simplification used in obtaining (3.37). A more accurate result for this variance can be obtained by avoiding this simplification. However, the derivation in this case becomes, if not impossible, extremely difficult. A consequence of this result is that the excess MSE of the algorithm turns out to

be zero, as is shown in Section 3.5. This result, however, has been verified through simulations, indicating that the approximations are valid.

3.4. Discussion

Before we proceed further with the analysis of the RI algorithm, it is important to compare the LMS and the RI algorithms in terms of their respective update equations. This sheds light on the superior performance of the RI algorithm.

The filter tap update equation of the LMS algorithm as defined in [1] is,

$$\mathbf{w}(k) = \mathbf{w}(k-1) + \mu e(k)\mathbf{x}(k), \tag{3.41}$$

where $e(k)$ is given in (3.7)

Substituting (3.7) in (3.41) gives,

$$\mathbf{w}(k) = \left[\mathbf{I} - \mu\mathbf{x}(k)\mathbf{x}^T(k)\right]\mathbf{w}(k-1) + \mu\mathbf{x}(k)d(k), \tag{3.42}$$

where,

$$\mathbf{x}(k)d(k) = \bar{\mathbf{p}} + \delta\mathbf{p}(k), \tag{3.43}$$

where $\bar{\mathbf{p}} = E\left\{\mathbf{x}(k)d(k)\right\}$.

Now, substituting (3.25), (3.19) and (3.43) in (3.42) gives,

$$\bar{\mathbf{w}}_k + \delta\widetilde{\mathbf{w}}_k = \left[\mathbf{I} - \mu\mathbf{R}_{xx} - \mu\delta\mathbf{R}(k)\right]\left(\bar{\mathbf{w}}_{k-1} + \delta\widetilde{\mathbf{w}}_{k-1}\right) + \mu\bar{\mathbf{p}} + \mu\delta\mathbf{p}(k). \tag{3.44}$$

Separating (3.44) into its deterministic and stochastic parts, respectively, results,

$$\bar{\mathbf{w}}_k = \left[\mathbf{I} - \mu\mathbf{R}_{xx}\right]\bar{\mathbf{w}}_{k-1} + \mu\bar{\mathbf{p}}, \tag{3.45}$$

and

$$\delta\widetilde{\mathbf{w}}_k = \left[\mathbf{I} - \mu\mathbf{x}(k)\mathbf{x}^T(k)\right]\delta\widetilde{\mathbf{w}}_{k-1} + \mu\left[\delta\mathbf{p}(k) - \delta\mathbf{R}(k)\bar{\mathbf{w}}_{k-1}\right]. \tag{3.46}$$

Now, let us rearrange (3.45) and (3.46), and compare them with (3.27) and (3.28) as shown in Table 3.1.

Table 3.1. Comparison of the stochastic update equations of the RI and LMS algorithms

LMS algorithm	RI algorithm
$\bar{\mathbf{w}}_k = \left[\mathbf{I} - \mu\mathbf{R}_{xx}\right]\bar{\mathbf{w}}_{k-1} + \mu\bar{\mathbf{p}}$	$\bar{\mathbf{w}}_k = \left[\mathbf{I} - \mu_0\mathbf{R}_{xx}\right]\bar{\mathbf{w}}_{k-1} + \mu_0\bar{\mathbf{p}}$
$\delta\widetilde{\mathbf{w}}_k = \left[\mathbf{I} - \mu\left(\mathbf{R}_{xx} + \delta\mathbf{R}_k\right)\right]\delta\widetilde{\mathbf{w}}_{k-1}$ $+\mu\left[\delta\mathbf{p}_k - \delta\mathbf{R}_k\bar{\mathbf{w}}_{k-1}\right]$	$\delta\widetilde{\mathbf{w}}_k = \left[\mathbf{I} - \mu_0\left(\mathbf{R}_{xx} + \frac{1}{\mu_0}\delta\widetilde{\mathbf{A}}_k\right)\right]\delta\widetilde{\mathbf{w}}_{k-1}$ $+\left[\delta\widetilde{\mathbf{p}}_k - \delta\widetilde{\mathbf{A}}_k\bar{\mathbf{w}}_{k-1}\right]$

Observations:

1. In the equation of the random part of $\mathbf{w}_k$, $\delta\widetilde{\mathbf{A}}_k$ is a low-pass filtered version of the covariance matrix random part $\delta\mathbf{R}_k$ (with a small cut-off frequency). This implies that, the eigenvalues of the instantaneous estimate of the covariance matrix, $\mathbf{x}_k\mathbf{x}_k^T = \mathbf{R}_{xx} + \delta\mathbf{R}_k$, vary in a much wider range than those of $\mathbf{R}_{xx} + \frac{1}{\mu_0}\delta\widetilde{\mathbf{A}}_k$; i.e. $\lambda_{max}(k)$ of $\mathbf{x}_k\mathbf{x}_k^T$ is expected to be much larger than $\lambda_{max}(k)$ of

$\mathbf{R}_{xx} + \frac{1}{\mu_0}\delta\widetilde{\mathbf{A}}_k$. Therefore, step-size μ should be chosen much smaller than μ_0 (i.e. convergence of the equation of $\bar{\mathbf{w}}_k$ is much slower in the LMS algorithm).

2. The forcing term in the equation of $\delta\widetilde{\mathbf{w}}_k$ in the RI algorithm is again a low-pass filtered version of that in the LMS algorithm. Hence, the fluctuations of $\delta\widetilde{\mathbf{w}}_k$ may be expected to be much smaller than those of $\delta\widetilde{\mathbf{w}}_k$ in the LMS algorithm.

3. The update equations for the mean weight vector are identical. However, as pointed out in observation 1, the possibility of choosing a much larger value of the step-size in the RI algorithm is a major advantage over the LMS algorithm. Although the convergence of the RI algorithm is also dependent on the eigenvalue spread of the autocorrelation matrix, the facts pointed out in observation 1 imply that the situation is much worse in the LMS algorithm because of the larger fluctuations of the eigenvalues. Another important fact that must be kept in mind is that even with an eigenvalue spread approaching unity, the RLS algorithm cannot converge very fast because of the slowness of the recursions for the correlations in (3.2) and (3.3) unless β is considerably smaller than unity; a case in which the performance of the RLS deteriorates. This means that an eigenvalue spread to some degree is allowable; such that convergence of the slowest mode of the weight vector is not slower than the convergence of the correlations. To show this, let the slowest mode be expressed as:

$$v_s(k) = (1 - \mu_0\lambda_{min})^k\,, \tag{3.47}$$

where μ_0 is chosen to satisfy the stability criterion $\left(\mu_0 < \frac{2}{\lambda_{max}(\mathbf{R}_{xx})}\right)$. Letting $\mu_0\lambda_{max} = 2\eta$, where $\eta < 1$, and $\chi(\mathbf{R}_{xx}) = \frac{\lambda_{max}}{\lambda_{min}}$, then the slowest mode becomes

$$v_s(k) = \left(1 - \frac{2\eta}{\chi(\mathbf{R}_{xx})}\right)^k. \tag{3.48}$$

On the other hand, the convergence of the correlations is determined by the factor β^k. Therefore, the mean weights will converge faster than the correlations if

$$\left|1 - \frac{2\eta}{\chi(\mathbf{R}_{xx})}\right| < \beta < 1. \tag{3.49}$$

For a large eigenvalue spread $(\chi(\mathbf{R}_{xx}))$ and η of the order of 1, β will be slightly less than unity, which is in agreement with the usual choice in the RLS algorithm.

3.5. The Ensemble - Average Learning Curve of the RI Algorithm

The computation of the ensemble-average learning curve of the RI algorithm is based on the *a priori* estimation error $e(k)$. Now, define:

$$J(k) = E\left\{|e(k)|^2\right\}, \tag{3.50}$$

where $J(k)$ is the mean-square-error value of $e(k)$, and $e(k)$ is the *a priori* filtering error given by (3.7).

The desired signal $d(k)$ can be modeled as [1]:

$$d(k) = e_0(k) + \mathbf{w}_o^T\mathbf{x}(k), \tag{3.51}$$

where $e_0(k)$ is the measurement noise.

Now, substituting (3.51) in (3.7) yields;

$$\begin{aligned} e(k) &= e_0(k) + [\mathbf{w}_o - \mathbf{w}(k-1)]^T \mathbf{x}(k) \\ &= e_0(k) + \varepsilon^T(k-1)\mathbf{x}(k), \end{aligned} \tag{3.52}$$

where $\varepsilon(k-1)$ is the undisturbed estimation error and is equal to;

$$\varepsilon(k-1) = \mathbf{w}_o - \mathbf{w}(k-1). \tag{3.53}$$

Substituting (3.52) in (3.50) gives:

$$J(k) = E\{\left[e_0(k) + \varepsilon^T(k-1)\mathbf{x}(k)\right]\left[e_0(k) + \varepsilon^T(k-1)\mathbf{x}(k)\right]^T\}.$$

Simplification of the above equation yields

$$\begin{aligned} J(k) &= E\left\{|e_0(k)|^2\right\} + E\left\{e_0(k)\mathbf{x}^T(k)\varepsilon(k-1)\right\} + E\left\{e_0(k)\varepsilon^T(k-1)\mathbf{x}(k)\right\} \\ &+ E\left\{\varepsilon^T(k-1)\mathbf{x}(k)\mathbf{x}^T(k)\varepsilon(k-1)\right\}. \end{aligned} \tag{3.54}$$

The second and third terms of (3.54) are equal to zero for two reasons. First, the undisturbed estimation error (weight-error vector) $\varepsilon(k-1)$ depends on the past values of the input vector $\mathbf{x}(k)$ and the measurement noise $e_0(k)$. Second, $\mathbf{x}(k)$ and $e_0(k)$ are statistically independent and $e_0(k)$ has a zero mean. Hence,

$$E\left\{e_0(k)\mathbf{x}^T(k)\varepsilon(k-1)\right\} = E\left\{e_0(k)\varepsilon^T(k-1)\mathbf{x}(k)\right\}$$
$$= 0. \tag{3.55}$$

Assuming that $e_0(k)$ is a white noise process with zero mean and variance σ_0^2, the first term of (3.54) becomes:

$$E\left\{|e_0(k)|^2\right\} = E\left\{e_0(i)e_0^*(j)\right\} = \begin{cases} \sigma_0^2, & i = j \\ 0, & i \neq j. \end{cases} \tag{3.56}$$

Now, consider the last term in (3.54),

$$E\left\{\varepsilon^T(k-1)\mathbf{x}(k)\mathbf{x}^T(k)\varepsilon(k-1)\right\} = E\left\{tr\left[\mathbf{x}(k)\mathbf{x}^T(k)\varepsilon(k-1)\varepsilon^T(k-1)\right]\right\}$$
$$= tr\left[E\left\{\mathbf{x}(k)\mathbf{x}^T(k)\varepsilon(k-1)\varepsilon^T(k-1)\right\}\right] \tag{3.57}$$

Assuming that the fluctuations in the weight-error vector $\varepsilon(k)$ are slower compared with those of the input signal vector $\mathbf{x}(k)$ [1, p.449], and by using the direct averaging method (The direct averaging technique described in [83] means that the term $\mathbf{x}(k)\mathbf{x}^T(k)$ is substituted with the expectation over the ensemble. Hence, $\mathbf{x}(k)\mathbf{x}^T(k) = E\left[\mathbf{x}(k)\mathbf{x}^T(k)\right] = \mathbf{R}_{xx}$), (3.57) can be rewritten as:

$$tr\left[E\left\{\mathbf{x}(k)\mathbf{x}^T(k)\varepsilon(k-1)\varepsilon^T(k-1)\right\}\right] \approx tr\left[E\left\{\mathbf{x}(k)\mathbf{x}^T(k)\right\}E\left\{\varepsilon(k-1)\varepsilon^T(k-1)\right\}\right]$$
$$= tr\left[\mathbf{R}_{xx}\mathbf{K}(k-1)\right], \tag{3.58}$$

where

$$\mathbf{K}(k) = E\left\{\varepsilon(k)\varepsilon^T(k)\right\}. \tag{3.59}$$

Now, substituting (3.53) in (3.59) gives:

$$\mathbf{K}(k) = E\left\{\left[\mathbf{w}(k) - \mathbf{w}_o\right]\left[\mathbf{w}(k) - \mathbf{w}_o\right]^T\right\}, \tag{3.60}$$

Substituting (3.25) and (3.29) in (3.60) gives:

$$\begin{aligned}
\mathbf{K}(k) &= E\left\{\left[\Delta\mathbf{w}(k) + \delta\widetilde{\mathbf{w}}(k)\right]\left[\Delta\mathbf{w}(k) + \delta\widetilde{\mathbf{w}}(k)\right]^T\right\} \\
&= E\left\{\Delta\mathbf{w}(k)\Delta\mathbf{w}^T(k)\right\} + E\left\{\Delta\mathbf{w}(k)\delta\widetilde{\mathbf{w}}^T(k)\right\} \\
&+ E\left\{\delta\widetilde{\mathbf{w}}(k)\Delta\mathbf{w}^T(k)\right\} + E\left\{\delta\widetilde{\mathbf{w}}(k)\delta\widetilde{\mathbf{w}}^T(k)\right\},
\end{aligned} \tag{3.61}$$

The second and third terms of (3.61) are zero by virtue of independence between $\Delta\mathbf{w}(k)$ and $\delta\widetilde{\mathbf{w}}(k)$, and by the definition of $\delta\widetilde{\mathbf{w}}(k)$ as a zero mean perturbation, $(E\left\{\delta\widetilde{\mathbf{w}}(k)\right\} = 0)$.

Now, taking the first term of (3.61) and by using the result of (3.32), yields,

$$\begin{aligned}
E\left\{\Delta\mathbf{w}(k)\Delta\mathbf{w}^T(k)\right\} &= E\left\{\left[\mathbf{I} - \mu_0\mathbf{R}_{xx}\right]^k \Delta\mathbf{w}(0)\Delta\mathbf{w}^T(0)\left[\mathbf{I} - \mu_0\mathbf{R}_{xx}\right]^k\right\} \\
&= \left[\mathbf{I} - \mu_0\mathbf{R}_{xx}\right]^k \mathbf{Z}_0 \left[\mathbf{I} - \mu_0\mathbf{R}_{xx}\right]^k,
\end{aligned} \tag{3.62}$$

where $\mathbf{Z}_0 = E\left\{\Delta\mathbf{w}(0)\Delta\mathbf{w}^T(0)\right\}$.

Now, substituting (3.40) and (3.62), in (3.61) gives:

$$\mathbf{K}(k) = \left[\mathbf{I} - \mu_0 \mathbf{R}_{xx}\right]^k \mathbf{K}_0 \left[\mathbf{I} - \mu_0 \mathbf{R}_{xx}\right]^k, \tag{3.63}$$

where $\mathbf{K}_0 = E\left\{\left[\Delta\mathbf{w}(0) + \delta\widetilde{\mathbf{w}}(0)\right]\left[\Delta\mathbf{w}(0) + \delta\widetilde{\mathbf{w}}(0)\right]^T\right\} = \mathbf{Z}_0 + \mathbf{Q}_0 = E\left\{\mathbf{w}_o\mathbf{w}_o^T\right\} = \mathbf{w}_o\mathbf{w}_o^T$. By using the eigenvalue decomposition (EVD) of $\mathbf{R}_{xx}$, $\mathbf{R}_{xx} = \mathbf{S}\Lambda\mathbf{S}^T$, in (3.63),

$$\mathbf{K}(k) = \mathbf{S}\left[\mathbf{I} - \mu_0\Lambda\right]^k \mathbf{S}^T\mathbf{K}_0\mathbf{S}\left[\mathbf{I} - \mu_0\Lambda\right]^k \mathbf{S}^T, \tag{3.64}$$

where Λ and $\mathbf{S}$ are matrices having the eigenvalues and eigenvectors of $\mathbf{R}_{xx}$, respectively.

Substituting (3.64) in (3.58) and by diagonalizing $\mathbf{R}_{xx}$;

$$\begin{aligned} tr\left[\mathbf{R}_{xx}\mathbf{K}(k-1)\right] &= tr\left[\mathbf{S}\Lambda\mathbf{S}^T\mathbf{S}\left[\mathbf{I} - \mu_0\Lambda\right]^k \mathbf{S}^T\mathbf{K}_0\mathbf{S}\left[\mathbf{I} - \mu_0\Lambda\right]^k \mathbf{S}^T\right] \\ &= tr\left[\Lambda\left[\mathbf{I} - \mu_0\Lambda\right]^k \mathbf{S}^T\mathbf{K}_0\mathbf{S}\left[\mathbf{I} - \mu_0\Lambda\right]^k\right], \end{aligned} \tag{3.65}$$

where the facts that $\mathbf{S}^T\mathbf{S} = \mathbf{I}$ and $tr(\mathbf{AB}) = tr(\mathbf{BA})$ are used. Substituting this result in (3.65);

$$\begin{aligned} tr\left[\mathbf{R}_{xx}\mathbf{K}(k-1)\right] &= tr\left[\mathbf{S}^T\mathbf{K}_0\mathbf{S}\left[\mathbf{I} - \mu_0\Lambda\right]^{2k}\Lambda\right] \\ &= \sum_{i=1}^{N} \zeta_i \left[1 - \mu_0\lambda_i\right]^{2k} \lambda_i. \end{aligned} \tag{3.66}$$

where ζ_i are the diagonal elements of the matrix $\mathbf{S}^T\mathbf{K}_0\mathbf{S}$. Substituting (3.56) and (3.66) in (3.54),

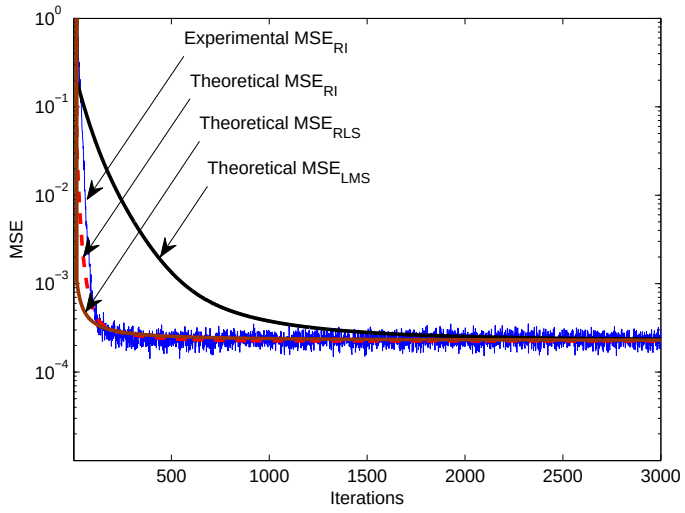

Figure 3.1. The MSE learning curves of the RI, RLS and LMS algorithms.

$$J(k) = \sigma_0^2 + \sum_{i=1}^{N} \zeta_i \lambda_i \left[1 - \mu_0 \lambda_i\right]^{2k}. \tag{3.67}$$

In equation (3.67), with a proper selection of μ_0 based on the discussions in section 3.4, the second term can be made to vanish in a number of time steps comparable to that of RLS, and much smaller than that of LMS. Equation (3.67) implies that the excess MSE of the RI algorithm is zero. Fig. 3.1 shows that, for the system identification problem described in Chapter 4 Section 4.3, the theoretical and experimental MSE curves for the RI algorithm are in good agreement. The differences between the two MSE curves are the consequence of the approximations made in deriving (3.67). Keeping the same MSE constant for the other algorithms, it also shows that the RI algorithm converges almost at the same time as the RLS algorithm and much faster than that of the LMS algorithm.

3.6. Fast Implementation Issues of the RI Algorithm

From equations (3.2), (3.3) and (3.16), it is noted that the RI algorithm requires $\frac{5}{2}N^2 + \frac{7}{2}N$ floating point multiplications and divisions, and $2N^2 + N$ floating point additions and subtractions. Even though the number of computations is less than that of the RLS algorithm, it can be reduced even further.

The main idea is to obtain a Toeplitz approximation of $\mathbf{R}(k)$ (see appendix) and apply transform techniques to carry out the multiplication $\mathbf{R}(k)\mathbf{w}(k-1)$ in the update equation. Now, considering (3.2), the elements of $\mathbf{R}(k)$ can be updated using,

$$r_{i,j}(k) = \beta r_{i,j}(k-1) + x_i(k)x_j(k), \tag{3.68}$$

where $i = 1, 2, \ldots, N-q$, $j = i+q$ and $q = 0, 1, \ldots, N-1$. The correlations in the q^{th} diagonal of the autocorrelation estimate, with $q = 0$ corresponding to the main diagonal, can be averaged using (3.68) as:

$$r_q(k) = \beta r_q(k-1) + \frac{1}{N-q}\sum_{i=1}^{N-q} x_i(k)x_j(k), \tag{3.69}$$

where,

$$r_q(k) = \frac{1}{N-q}\sum_{i=1}^{N-q} r_{i,j}(k).$$

By (3.69) the symmetric sequence $g_q(k)$ can be constructed as:

$$g_q(k)=\begin{cases} r_q, & 0 \leq q \leq (N-1) \\ r_q^*, & -(N-1) \leq q < 0 \end{cases} \tag{3.70}$$

where $r_q^* = r_{-q}$.

The n^{th} element of the vector $\mathbf{w}_f(k) = \mathbf{R}(k)\mathbf{w}(k)$ can be written as:

$$w_{f,n}(k) = \sum_{m=1}^{N} r_{n,m} w_{m-1}(k), \quad n = 1, 2, \ldots, N. \tag{3.71}$$

Rewriting (3.71) in terms of the sequence in (3.70) gives

$$w_{f,n}(k) = \sum_{m=0}^{N-1} g_{n-m-1} w_m(k), \quad n = 1, 2, \ldots, N, \tag{3.72}$$

Equation (3.72) represents the convolution sum. Now, taking $(2N-1)$-point DFT of both sides of (3.72) at time k

$$W_{fe}(l) = G(l)W_e(l), \quad l = 1, 2, \ldots, 2N-1, \tag{3.73}$$

where $W_{fe}(l)$ is the DFT of the zero-padded sequence $\{w_{fe,n}(k); \quad n = 1, 2, \ldots, 2N-1\}$:

$$w_{fe,n}(k)=\begin{cases} w_{f,n}(k), & n = 1, 2, \ldots, N \\ 0, & n = N+1, \ldots, 2N-1, \end{cases} \tag{3.74}$$

and $W_e(l)$ is the DFT of $\mathbf{w}_e(k) = [\underline{\mathbf{0}} \ \ \mathbf{w}(k)]$ where $\underline{\mathbf{0}}$ is an $(N-1)$-dimensional zero vector. The sequence $\{w_{f,n}(k); \quad n = 1, 2, \ldots, N\}$ can now be recovered from the

inverse DFT of $W_{fe}(l)$.

By applying this method, the computational complexity of the RI algorithm will be significantly reduced as shown in Table 3.2 and Fig. 3.2. It can be observed here that the RI, Fast RI and the RLS algorithms all have complexity $O(N^2)$. But Fig. 3.2 shows that as the filter length increases the difference in computational complexity becomes more prominent.

Table 3.2. Computational Complexity of RI, fast RI, 2^{nd} order RI, RLS and SFTRLS algorithms.

	Mult./Div.	Add./Sub.
RI	$\frac{5}{2}N^2 + \frac{7}{2}N$	$2N^2 + N$
RI$_{fast}$	$\frac{1}{2}N^2 + N\left(\frac{11}{2} + 3\log_2 N\right) - 1$	$\frac{1}{2}N^2 + N\left(\frac{3}{2} + 9\log_2 N\right) + 1$
2^{nd} order RI	$\frac{5}{2}N^2 + \frac{7}{2}N$	$\frac{5}{2}N^2 + 2N$
RLS	$3N^2 + 11N + 9$	$3N^2 + 7N + 4$
SFTRLS	$9N + 13$	$9N + 1$

3.7. Robust RI Algorithm in Impulsive Noise

Randomness of the input signal of adaptive filters is one of the factors that critically affect their convergence performance [15]. The randomness of the input signal caused by impulsive noise can cause the performance of the adaptive filter to deteriorate [8]. The RLS algorithm [1] has shown good performance, even in impulsive noise environments. However, the RLS algorithm suffers from its high computational complexity, stability problems when the forgetting factor is relatively low, and tracking capability when the forgetting factor is relatively high. The RI algorithm [50]-[53], was proposed to overcome these problems. However, the RLS and RI algorithms both provide a poor performance in impulsive noise environments

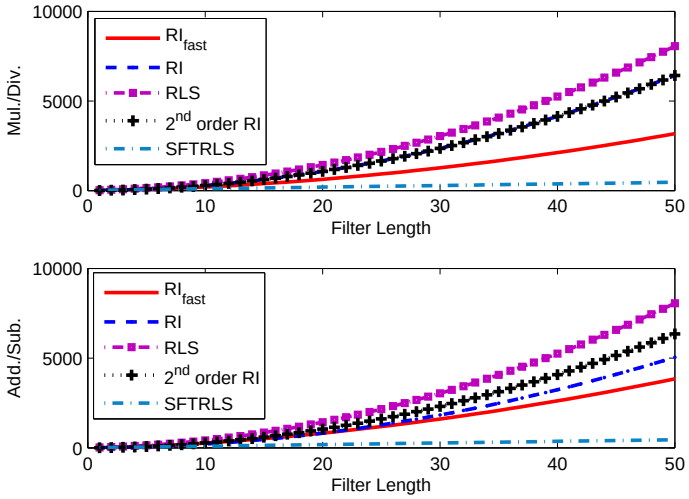

Figure 3.2. The Computational Complexity of the fast RI, RI, 2^{nd} order RI, RLS and SFTRLS algorithms.

when the signal-to-noise (SNR) is low. Several robust versions of the RLS algorithm for impulsive noise environments have been proposed, [17], [47]-[49]. These algorithms suffer from their high computational complexity, and stability problems due to the estimate of the inverse of the autocorrelation matrix [8].

In this section, we propose a new RI algorithm that is robust to impulsive noise and provides better mean-square-error (MSE) performance than the recently proposed Robust RLS algorithm [17], with a considerable reduction in the computational complexity [50] [51] as shown in Table 3.2. The proposed method employs the technique introduced in [17].

Now, let us consider estimating the autocorrelation matrix and cross-correlation vector as [17] instead of those in (3.2) and (3.3),

$$\mathbf{R}(k) = \beta\mathbf{R}(k-1) + \delta(k)\mathbf{x}(k)\mathbf{x}^T(k), \tag{3.75}$$

and

$$\mathbf{p}(k) = \beta\mathbf{p}(k-1) + \delta(k)d(k)\mathbf{x}(k), \tag{3.76}$$

where $\delta(k)$ is a nonnegative scalar given by,

$$\delta(k) = \frac{1}{\|\mathbf{x}(k)e(k)\|_1}. \tag{3.77}$$

After convergence, $e(k)$ may become arbitrarily close to zero (within the precision of the simulation software), yet the algorithm does not diverge. It is also observed that, $\delta(k)$ can be greater than unity which would affect the convergence performance of the adaptive filter. To overcome this problem [17], $\delta(k)$ is modified as

$$\delta(k) = \min\left(1, \frac{1}{\|\mathbf{x}(k)e(k)\|_1}\right), \tag{3.78}$$

The L_1 norm of the gain vector, $\mathbf{p}(k) - \beta\mathbf{p}(k-1)$, in impulsive noise environments which is given by

$$\|\mathbf{p}(k) - \beta\mathbf{p}(k-1)\|_1 = \|\delta(k)d(k)\mathbf{x}(k)\|_1, \tag{3.79}$$

increases suddenly when $d(k)$ is corrupted by impulsive noise. Hence, the L_1 norm of $\mathbf{p}(k)$ also increases, and this, in turn, would increase the L_1 norm of $\mathbf{w}(k)$ in (3.16). The effect of this impulsive noise can be suppressed by $\delta(k)$ which imposes a time-varying upper bound on the L_1 norm of the gain vector in (3.79).

With $\delta(k) = 1$, the update equations in (3.75) and (3.76) become identical to those of the original RI algorithm given in (3.2) and (3.3). Also, as it can be seen from (3.75), $\delta(k)$ bounds the L_1 norm of the gain matrix, $\mathbf{R}(k) - \beta\mathbf{R}(k-1)$, as

$$\begin{aligned} \|\mathbf{R}(k) - \beta\mathbf{R}(k-1)\|_1 &= \left\|\delta(k)\mathbf{x}(k)\mathbf{x}^T(k)\right\|_1 \\ &= \min\left(\|\mathbf{x}(k)\|_\infty \|\mathbf{x}(k)\|_1, \frac{\|\mathbf{x}(k)\|_\infty}{|e(k)|}\right). \end{aligned} \quad (3.80)$$

Equations (3.78) and (3.80) prevent $e(k)$ from affecting $\delta(k)$ and then the L_1 norm of the gain matrix. It can be further deduced from (3.80) that, the L_1 norm of the gain matrix would be reduced. Since the probability of having $\delta(k) = 1$ is high in the transient state and the convergence of the RI algorithm is fast, the convergence of the proposed robust RI algorithm will also be fast.

When an impulsive noise-corrupted $e(k)$ occurs, we obtain $|d(k)| \approx |e(k)|$ and $\delta(k) = \frac{1}{\|\mathbf{x}(k)e(k)\|_1}$ which would enforce the L_1 norm of the gain vector in (3.79) and the L_1 norm of the gain matrix in (3.80) to be bounded by unity and $\frac{\|\mathbf{x}(k)\|_\infty}{|e(k)|}$, respectively. Hence the L_1 norm of the coefficient vector in (3.16) would also be bounded.

The differential coefficient vector of the RI algorithm is given by

$$\begin{aligned} \Delta\mathbf{w}(k) &= \mathbf{w}(k) - \mathbf{w}(k-1) \\ &= -\mu(k)\mathbf{R}(k)\mathbf{w}(k-1) + \mu(k)\mathbf{p}(k). \end{aligned} \quad (3.81)$$

In the steady state, assuming that $\mathbf{p}(k) = \mathbf{R}(k)\mathbf{w}(k)$, then substituting (3.2) and (3.3) in (3.81) and rearranging yields

$$\Delta \mathbf{w}(k) = \mu(k)\mathbf{x}(k)\left(d(k) - \mathbf{x}^T(k)\mathbf{w}(k-1)\right). \tag{3.82}$$

Substituting (3.7) in (3.82) gives

$$\Delta \mathbf{w}(k) = \mu(k)\mathbf{x}(k)e(k). \tag{3.83}$$

The L_1 norm of the differential coefficient vector of the RI algorithm is

$$\|\Delta \mathbf{w}(k)\|_1 = |\mu(k)| \, \|\mathbf{x}(k)e(k)\|_1 . \tag{3.84}$$

It is seen from (3.84) that, the L_1 norm of the differential coefficient vector of the RI algorithm may increase abruptly in the steady state for an impulsive-noise-corrupted $e(k)$. Similarly, using (3.75) and (3.76), we get the L_1 norm of the differential coefficient vector of the proposed robust RI algorithm as

$$\|\Delta \mathbf{w}(k)\|_1 = |\mu(k)| \, |\delta(k)| \, \|\mathbf{x}(k)e(k)\|_1 . \tag{3.85}$$

Substituting (3.77) in (3.85) gives

$$\|\Delta \mathbf{w}(k)\|_1 = |\mu(k)| . \tag{3.86}$$

In other words, the proposed robust RI algorithm provides robust performance with respect to a long burst of impulsive noise. Using the well known vector-norm inequality

$$\frac{1}{\sqrt{N}} \|\Delta \mathbf{w}(k)\|_1 \leq \|\Delta \mathbf{w}(k)\|_2 \leq \|\Delta \mathbf{w}(k)\|_1 , \tag{3.87}$$

we note that the L_2 norm of the differential-coefficient vector would also remain bounded and hence the norm of $\mathbf{w}(k)$ in the proposed robust RI algorithm would also be robust with respect to the impulsive-noise-corrupted $e(k)$.

3.8. The Effect of the Forgetting Factor on the RI Adaptive Algorithm in System Identification

In the RLS algorithm, the forgetting factor is chosen between zero and unity. However, there is a compromise between the performance criteria depending on the value of this forgetting factor [64]: When the forgetting factor is very close to unity, the RLS algorithm achieves good stability, whereas its tracking capabilities are reduced. A smaller value of the forgetting factor improves the tracking capabilities of the RLS algorithm but it decreases its probability of being stable [1].

Let us consider a system identification setting [1] as shown in Fig. 3.3. The output of the unknown system is assumed to be corrupted by an additive white Gaussian noise (AWGN). In this scenario, the aim of the adaptive filter is to recover the system noise from the error signal after it converges to the true solution. However the error signal must not go to zero, since this may introduce noise in the adaptive filter. Moreover, the RLS algorithm has a feature which makes the problem even more complicated. Under some conditions, it is able to cancel the error of the adaptive filter even in the presence of system noise. In other words, the system noise *leaks* into the output of the adaptive filter, which of course may lead to an incorrect

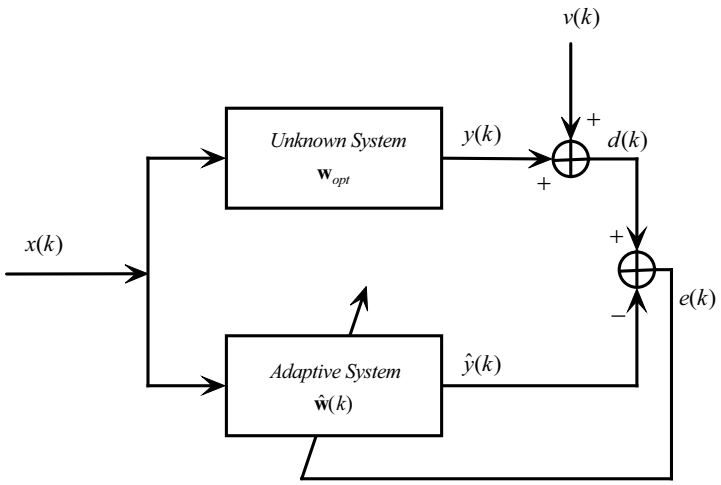

Figure 3.3. Block diagram of adaptive system identification.

solution to the problem. This *leakage* depends on the values of the forgetting factor and the length of the adaptive filter [43].

In this section, we analyze this *leakage* phenomenon for the RI algorithm [65], and give a theoretical estimation of it in terms of the forgetting factor β and the filter length N. These findings will be compared with those of the RLS algorithm introduced in [43].

3.8.1. The Leakage Phenomenon

3.8.1.1. System Identification - Ideal Behavior. In the system identification setting shown in Fig. 3.3, our objective is to estimate the unknown system coefficients using an adaptive filter, both driven by the same input signal $x(k)$. The unknown and the adaptive systems are assumed to be, both, Finite-Impulse-Response (FIR) filters of length N, defined by the tap-weight vectors

$$\mathbf{w} = [w_0 \; w_1 \; \ldots \; w_{N-1}]^T \, ,$$

and

$$\hat{\mathbf{w}}(k) = [\hat{w}_0(k)\ \hat{w}_1(k)\ \ldots\ \hat{w}_{N-1}(k)]^T\,,$$

As seen from Fig. 3.2, $y(k)$, the output of the unknown system is given by

$$y(k) = \mathbf{x}^T(k)\mathbf{w}, \tag{3.88}$$

where $\mathbf{x}(k)$ is the tap-input vector that contains the N most recent samples of the input signal given as,

$$\mathbf{x}(k) = [x(k)\ x(k-1)\ \ldots\ x(k-N+1)]^T\,.$$

The desired response $d(k)$ is defined as,

$$d(k) = y(k) + \nu(k) = \mathbf{x}^T(k)\mathbf{w} + \nu(k), \tag{3.89}$$

where $\nu(k)$ is the system (measurement) noise that corrupts the output of the unknown system, $y(k)$.

According to the block diagram in Fig. 3.2, the error signal $e(k)$ is

$$e(k) = [y(k) - \hat{y}(k)] + \nu(k). \tag{3.90}$$

From (1),

$$\mathbf{R}\hat{\mathbf{w}} = \mathbf{p}, \tag{3.91}$$

where $\mathbf{R}$ and $\mathbf{p}$ are the expected values of the autocorrelation matrix of the tap-input vector and cross-correlation vector between the desired output signal and the tap-input vector, respectively, and given by

$$\mathbf{R} = E\left\{\mathbf{x}(k)\mathbf{x}^T(k)\right\}, \tag{3.92}$$

$$\mathbf{p} = E\left\{\mathbf{x}(k)d(k)\right\}. \tag{3.93}$$

Substituting (3.89) in (3.93) gives

$$\begin{aligned}\mathbf{p} &= E\left\{\mathbf{x}(k)d(k)\right\} \\ &= E\left\{\mathbf{x}(k)\left[y(k)+\nu(k)\right]\right\} \\ &= E\left\{\mathbf{x}(k)y(k)\right\} + E\left\{\mathbf{x}(k)\nu(k)\right\},\end{aligned} \tag{3.94}$$

assuming that the input $\mathbf{x}(k)$ and the system noise $\nu(k)$ are statistically independent, then the second term of (3.94) will be eliminated and, therefore, substituting (3.88) in (3.94) gives

$$\begin{aligned}\mathbf{p} &= E\left\{\mathbf{x}(k)y(k)\right\} \\ &= E\left\{\mathbf{x}(k)\mathbf{x}^T(k)\mathbf{w}\right\} \\ &= \mathbf{R}\mathbf{w}.\end{aligned} \tag{3.95}$$

According to (3.91)

$$\hat{\mathbf{w}} = \mathbf{w}, \tag{3.96}$$

and

$$\hat{y}(k) = \mathbf{x}^T(k)\hat{\mathbf{w}} = \mathbf{x}^T(k)\mathbf{w} = y(k). \tag{3.97}$$

Substituting the result of (3.97) in (3.90) yields,

$$e(k) = \nu(k). \tag{3.98}$$

Or in other words, $y(k)$ and $\nu(k)$ are correctly separated.

3.8.1.2. System Identification - RI Algorithm. In the case of the RI algorithm, (3.92) and (3.93) are replaced by the recursive estimates of the instantaneous correlations given in (3.2) and (3.3).

Reconsidering (3.90),

$$\begin{aligned} e(k) = d(k) - \hat{y}(k) &= \mathbf{x}^T(k)\mathbf{w} + \nu(k) - \mathbf{x}^T(k)\mathbf{w}(k) \\ &= \mathbf{x}^T(k)\left[\mathbf{w} - \mathbf{w}(k)\right] + \nu(k), \end{aligned} \tag{3.99}$$

where $\mathbf{w}(k)$ is the recursive estimate of $\mathbf{w}$ given by (3.16) [50]-[53]:

As $k \to \infty$,

$$\mathbf{w}(k) = \mathbf{w}_0 + \delta\widetilde{\mathbf{w}}(k), \tag{3.100}$$

where $\mathbf{w}$ and $\delta\widetilde{\mathbf{w}}(k)$ are the optimum solution and the stochastic part of $\mathbf{w}(k)$, respectively.

Substituting (3.100) in (3.99) yields

$$e(k) = -\mathbf{x}^T(k)\delta\widetilde{\mathbf{w}}(k) + \nu(k), \tag{3.101}$$

For a good estimate of the unknown system, the difference between the output of the adaptive filter, $\hat{y}(k)$, and the output of the unknown system, $y(k)$, should approach zero in the steady state,

$$\begin{aligned} \hat{y}(k) - y(k) &= \hat{y}(k) - d(k) + \nu(k) \\ &= \hat{y}(k) - (e(k) + \hat{y}(k)) + \nu(k) \\ &= \nu(k) - e(k) = \mathbf{x}^T(k)\delta\widetilde{\mathbf{w}}(k). \end{aligned} \tag{3.102}$$

As a result of (3.102),

$$\hat{y}(k) = y(k) + \mathbf{x}^T(k)\delta\widetilde{\mathbf{w}}(k). \tag{3.103}$$

where $\delta\widetilde{\mathbf{w}}(k)$ [50] is given by

$$\delta\widetilde{\mathbf{w}}(k) = \left[\mathbf{I} - \mu_0\mathbf{R}_{xx}\right]\delta\widetilde{\mathbf{w}}(k-1) + \mu(k)\sum_{i=0}^{k}\beta^{k-i}\mathbf{x}(i)\nu(i). \tag{3.104}$$

From (3.103) and (3.104), we note that, in a system identification setting, the output of the adaptive filter, $\hat{y}(k)$, always contains a component which is proportional to the system noise, $\nu(k)$. This phenomenon is a leakage of the system noise, $\nu(k)$, in the output of the adaptive filter, $\hat{y}(k)$.

By (3.103), this leakage value is given as,

$$r(k) = \mathbf{x}^T(k)\delta\widetilde{\mathbf{w}}(k). \tag{3.105}$$

A normalized residual parameter [43], $\rho(k)$, may be defined as

$$\rho(k) = \left| \frac{\nu(k) - r(k)}{\nu(k)} \right|. \tag{3.106}$$

When total leakage occurs, then $\nu(k) = r(k)$ and $\rho(k) = 0$. This usually occurs at low values of the forgetting factor and high values of the filter length, as will be demonstrated in Chapter 4.

3.9. A 2-Dimensional (2D) RI Algorithm

Adaptive filtering techniques are very efficient in classical signal processing problems such as noise removal, channel estimation and system identification. Due to increased use of digital imaging and video in consumer electronics and multimedia applications, 2D adaptive filtering techniques gained more significance.

Approximately two decades ago, the LMS adaptive algorithm was extended from 1D to 2D [66], [67]. However, these algorithms update the filter coefficients only along the horizontal direction on a 2D plane. Consequently, these algorithms can

not sufficiently exploit information in 2D signals. This restriction is not a serious problem for stationary input signals, but it is serious for non-stationary input signals. For example, these algorithms result in degradations by the anisotropic processing when they process image signals [68].

Different types of 2D LMS adaptive algorithms, which can update the filter coefficients both along the horizontal and the vertical directions on a 2D plane were later developed, and applied to reduce noise in image signals [68]. One of the most efficient algorithms that has shown high performance in image noise removal and image deconvolution is the 2-D RLS algorithm [69], [70].

In this section, we propose a new 2D RI adaptive filtering technique [82] which provides a better performance than the 2D RLS algorithm [69], [70] in terms of Search Results peak signal-to-noise ratio (PSNR) [71] with reduced computational complexity.

3.9.1. Derivation of the 2D RI algorithm

Equation (3.16) describes the filter tap vector update equation of the RI algorithm. It can be generalized into its 2D form as:

$$\mathbf{w}_k(m_1, m_2) = [\mathbf{I} - \mu_k \mathbf{R}_k]\mathbf{w}_{k-1}(m_1, m_2) + \mu_k \mathbf{p}_k, \tag{3.107}$$

Where $\mathbf{w}_k(m_1, m_2)$ is the 2D tap-weight vector with dimensions $N \times N$, where $m_1 = 0, 1, \ldots, N-1$ and $m_2 = 0, 1, \ldots, N-1$, $\mathbf{R}_k$ and $\mathbf{p}_k$ are the instantaneous autocorrelation and cross-correlation matrices, respectively. They are estimated recursively as:

$$\mathbf{R}_k = \beta \mathbf{R}_{k-1} + \mathbf{x}(n_1, n_2)\mathbf{x}^T(n_1, n_2), \tag{3.108}$$

and

$$\mathbf{p}_k = \beta \mathbf{p}_{k-1} + d(n_1, n_2)\mathbf{x}(n_1, n_2). \tag{3.109}$$

Where $d(n_1, n_2)$ is the desired output, $\mathbf{x}(n_1, n_2)$ is the filter input. The filter input ($\mathbf{x}(n_1, n_2)$) and tap-weight vector ($\mathbf{w}_k(m_1, m_2)$) can be defined using the following column-ordered vectors,

$$\mathbf{x}(n_1, n_2) = \begin{bmatrix} x(n_1, n_2) \\ \vdots \\ x(n_1, n_2 - N + 1) \\ \vdots \\ x(n_1 - N + 1, n_2) \\ \vdots \\ x(n_1 - N + 1, n_2 - N + 1), \end{bmatrix} \tag{3.110}$$

$$\mathbf{w}_k(m_1, m_2) = \begin{bmatrix} w_k(0,0) \\ \vdots \\ w_k(0, N-1) \\ \vdots \\ w_k(N-1, 0) \\ \vdots \\ w_k(N-1, N-1), \end{bmatrix} \tag{3.111}$$

For 2D applications, there can be a number of ways that data can be reused [72]. One possible way is shown in Fig. 3.4. In this scheme, as shown in Fig. 3.4 (b), we consider a mask of 3×3 pixels which move horizontally to the right by one column at a time until the end of each row. Afterwards, the same process is repeated with the next row below until the last 9 pixels of the image are reached. At the end of each process of the mask, the data is reshaped as shown in Fig. 3.4 (a), starting from the last pixel in the lower right corner. The filter output is given by the following 2D convolution:

$$y(n_1, n_2) = \sum_{m_1=0}^{N-1} \sum_{m_2=0}^{N-1} w(m_1, m_2) x(n_1 - m_1, n_2 - m_2). \tag{3.112}$$

3.10. RI Adaptive Filter with Second Order Estimation of Autocorrelation Matrix

Even though the RI algorithm has a similar or better performance than the RLS algorithm most of the time, it should be noted that this performance may be improved further, with slight increment in the number of Add./Sub, by using the second order recursive updating of the correlations as i [73]

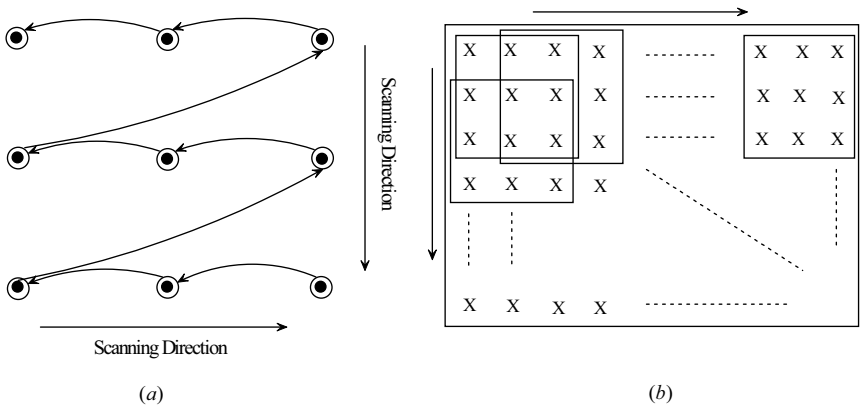

Figure 3.4. Rectangular Configuration of Data-Reusing in 2D.

$$\mathbf{R}(k) = \beta_1 \mathbf{R}(k-1) + \beta_2 \mathbf{R}(k-2) + \mathbf{x}(k)\mathbf{x}^T(k), \tag{3.113}$$

and

$$\mathbf{p}(k) = \beta_1 \mathbf{p}(k-1) + \beta_2 \mathbf{p}(k-2) + d(k)\mathbf{x}(k), \tag{3.114}$$

The number of multiplications in the second order equations will not be increased compared with the first order updating equations if the coefficients in (3.113) and (3.114) are chosen to be equal, i.e. $\beta_1 = \beta_2 = \frac{1}{2}\beta$.

Taking the expectation of (3.113) gives

$$\bar{\mathbf{R}}(k) = \frac{1}{2}\beta \bar{\mathbf{R}}(k-1) + \frac{1}{2}\beta \bar{\mathbf{R}}(k-2) + \mathbf{R}_{xx}, \tag{3.115}$$

where $\mathbf{R}_{xx} = E\left\{\mathbf{x}(k)\mathbf{x}^T(k)\right\}$ and $\bar{\mathbf{R}}(k) = E\{\mathbf{R}(k)\}$.

Poles of the system in (3.115) are:

$$\begin{aligned} z_1 &= \frac{1}{4}\left(\beta - \sqrt{\beta^2 + 8\beta}\right) \\ z_2 &= \frac{1}{4}\left(\beta + \sqrt{\beta^2 + 8\beta}\right) \end{aligned} \tag{3.116}$$

which have magnitudes less than unity if $\beta < 1$. Solving (3.115) with the initial conditions $\bar{\mathbf{R}}(-2) = \bar{\mathbf{R}}(-1) = \bar{\mathbf{R}}(0) = \mathbf{0}$ yields,

$$\bar{\mathbf{R}}(k) = \left(\frac{1}{\beta - 1} + \alpha_1 z_1^k + \alpha_2 z_2^k\right)\mathbf{R}_{xx}, \tag{3.117}$$

where

$$\begin{aligned} \alpha_1 &= \frac{\beta - z_2}{(1-\beta)(z_2 - z_1)} \\ \alpha_2 &= \frac{\beta - z_1}{(1-\beta)(z_2 - z_1)}. \end{aligned} \tag{3.118}$$

Letting,

$$\gamma(k) = \frac{1}{\beta - 1} + \alpha_1 z_1^k + \alpha_2 z_2^k, \tag{3.119}$$

then, in the RI algorithm, the variable step-size is chosen as:

$$\mu(k) = \frac{\mu_0}{\gamma(k)}. \tag{3.120}$$

The update equation of the tap weight vector will remain the same as in (3.16).

Chapter 4

SIMULATION RESULTS

4.1. Introduction

The performance of the RI algorithm and its variants mentioned in Chapter 3, is discussed in detail in this chapter.

In order to study the effect of noise distribution, we will discuss the Gaussian and the impulsive noise models used in this thesis when it is necessary. In the case of impulsive noise channels, increasing the impulsive component strength, or the probability of the outliers, will increase the noise power. This increment will affect the transmitted signal more.

In this thesis *MATLAB* Software Package is used for the simulation of the standard LMS, NLMS, VSSLMS, DCTLMS, TDVSS, RLS, SFTRLS, RRLS, PRRLS, PRI, fast RI, 2^{nd} order RI and 2D RI algorithms. Simulations are performed to investigate the performances of these algorithms in AWGN, correlated Gaussian noise and white and correlated impulsive noise for the following settings:

1. noise cancellation
2. system identification
3. channel equalization
4. echo cancellation

5. image deconvolution

in stationary and non-stationary environments.

4.2. Noise Cancellation

In this section, the performance of the Fast RI algorithm is compared to those of the RLS, VSSLMS and DCTLMS algorithms in a noise cancellation setting shown in Fig. 4.1. All the algorithms were implemented using an $\mathrm{SNR} = 10\log_{10}\left(\frac{P_{signal}}{P_{noise}}\right) = 30\mathrm{dB}$. The received signal was generated using [13]:

$$x(k) = 1.79x(k-1) - 1.85x(k-2) + 1.27x(k-3) - 0.41x(k-4) + v_0(k) \quad (4.1)$$

where $v_0(k)$ is a Gaussian process with zero mean and variance $\sigma^2 = 0.3849$.

4.2.1. Additive White Gaussian Noise

In this experiment, the signal is assumed to be corrupted with an Additive White Gaussian Noise (AWGN) process, with an overall eigenvalue spread $\chi(\mathbf{R}) = 230.75$; in all the experiments done, $\chi(\mathbf{R})$ is measured experimentally. Simulations were done with the following parameters: For the Fast RI algorithm: $\beta = 0.993$ and $\mu_0 = 0.00146$, where μ_0 is selected to be less than $\frac{2}{\lambda_{max}}$. As a result of property 8 [[1], p. 816], λ_{max} can be assumed to be less than or equal to the maximum power spectral density of the input signal ($\lambda_{max} \leq S_{max}$). Therefore, the maximum eigenvalue of the autocorrelation matrix can be estimated by computing the periodogram of the input signal. For the RLS algorithm: $\beta = 0.993$. For the VSSLMS algorithm: $\gamma = 0.00048$, $\alpha = 0.97$, $\mu_{min} = 0.0047$ and $\mu_{max} = 0.05$. For the DCTLMS algorithm: $\gamma = 0.002$, $\beta = 0.993$, $\epsilon = 8 \times 10^{-4}$, $\mu = 0.02$ and $M = 10$. Fig.

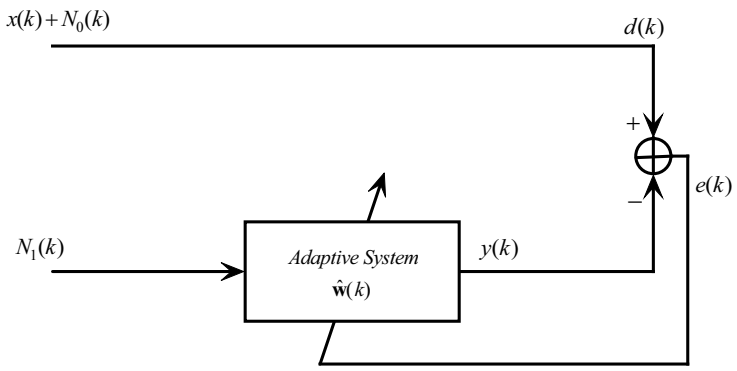

Figure 4.1. Block diagram of noise cancellation.

4.2 shows that the Fast RI and RLS algorithms converge to the same MSE almost at the same time, whereas the VSSLMS and DCTLMS algorithms converge to the same MSE but at a much slower rate than that of the Fast RI and RLS algorithms. This result indicates that, in the RI algorithm, with a proper choice of the step-size μ_0, as pointed out in chapter 3 section 3.5, the convergence rate can be made to approach that of the RLS algorithm, despite the very large eigenvalue spread of the autocorrelation matrix of the input signal.

It should be noted that the performance of the RI algorithm deteriorates when the eigenvalue spread is extremely large. To show the performance of the RI algorithm compared to those of the RLS, normalized LMS (NLMS) and LMS algorithms, let us consider an AR(1) process as follows:

$$x(k) = \rho x(k-1) + 0.4359r(k). \tag{4.2}$$

where ρ is the correlation coefficient and $r(k)$ is an AWGN process. When $\rho = 0.99$ the eigenvalue spread of the autocorrelation matrix is computed as $\chi(\mathbf{R}) = 4130$ on

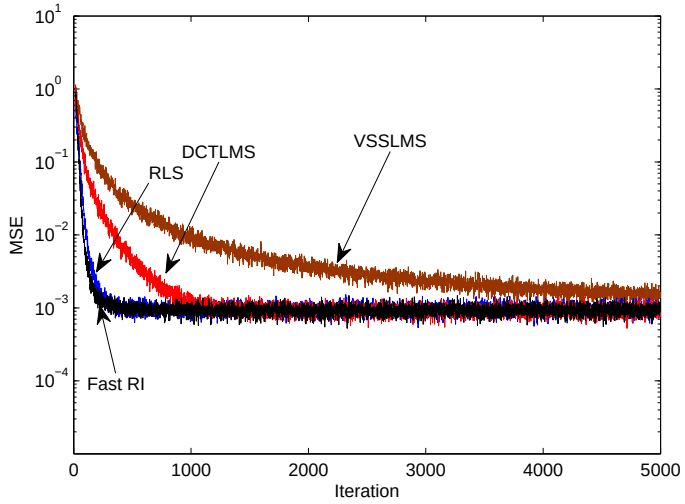

Figure 4.2. The ensemble MSE for the fast RI, RLS, VSSLMS and DCTLMS in AWGN.

the average, which is extremely high. Note that the theoretical eigenvalue spread is not valid in real-time due to the fluctuations of the estimated autocorrelation matrix. The algorithms were simulated with the following parameters: For the RI algorithm: $\beta = 0.993$ and $\mu_0 = 0.00001$. For the RLS algorithm: $\beta = 0.993$. For the NLMS algorithm: $\mu = 0.04$. For the LMS algorithm: $\mu = 0.00009$. Fig. 4.3 shows that the RI algorithm performance is worse than those of the RLS and NLMS algorithm and almost similar to that of LMS algorithm. This is due to the fact that, because of the very large eigenvalue ratio in (3.49) β is forced to be extremely close to unity, in which case the convergence of the correlations would be extremely slow. Hence if beta in the RI algorithm is chosen to be equal to that in RLS, the condition in (3.49) would be violated and the convergence behavior of the RI algorithm would be dominated by the eigenvalue ratio. In order to show that the convergence rate of the RI algorithm approaches that of RLS, we consider the same system with $\rho = 0.87$ (measured $\chi(\mathbf{R}) = 172$), and the algorithms were implemented with

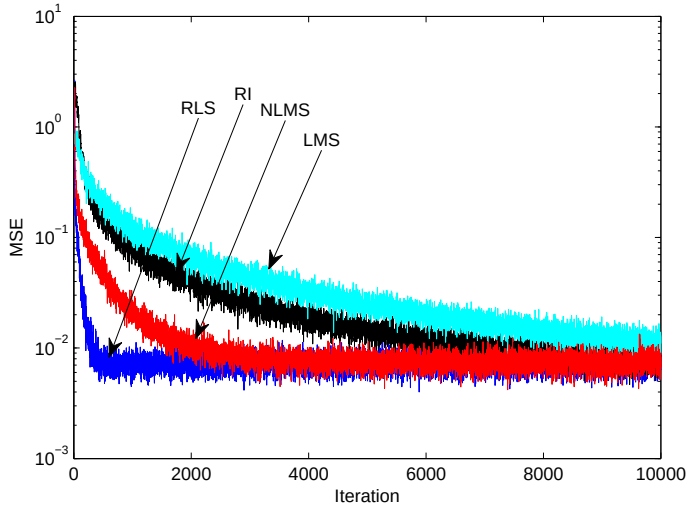

Figure 4.3. The ensemble MSE for the RI, RLS, NLMS and LMS in AWGN.

the parameters: For the RI algorithm: $\beta = 0.993$ and $\mu_0 = 0.0005$. For the RLS algorithm: $\beta = 0.993$. For the NLMS algorithm: $\mu = 0.04$. For the LMS algorithm: $\mu = 0.009$. Fig. 4.4 shows that the RI algorithm performs approximately the same as the RLS and much better than the NLMS and LMS algorithms. The advantage of the algorithm over the RLS algorithm is in terms of computational complexity.

It should be noted that the performance the RLS is sensitive to the forgetting factor β (β should be selected very close to unity) as mentioned before, where this is not the case in the RI algorithm. Taking $\rho = 0.90$ (measured $\chi(\mathbf{R}) = 399$), the algorithms were simulated with the following parameters: For the RI algorithm: $\beta = 0.97$ and $\mu_0 = 0.0007$. For the RLS algorithm: $\beta = 0.97$. For the NLMS algorithm: $\mu = 0.04$. For the LMS algorithm: $\mu = 0.009$. Fig. 4.5 shows that the RI algorithm converges to a lower MSE than that of the RLS algorithm (approximately 6 dB better) and converges faster than the NLMS and LMS algorithms.

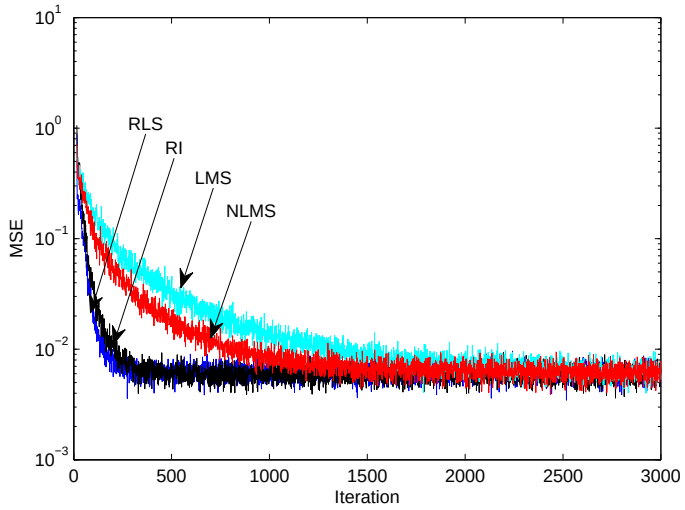

Figure 4.4. The ensemble MSE for the RI, RLS, NLMS and LMS in AWGN.

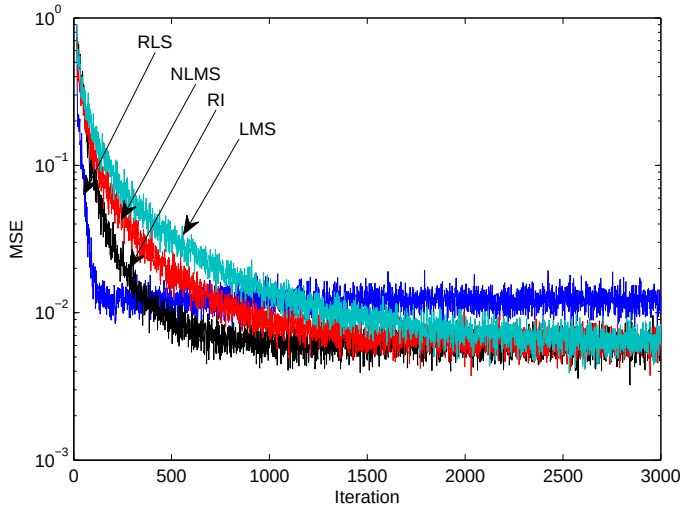

Figure 4.5. The ensemble MSE for the RI, RLS, NLMS and LMS in AWGN.

4.2.2. Additive Correlated Gaussian Noise

In this experiment, the signal is assumed to be corrupted with an Additive Correlated Gaussian Noise (ACGN) process, giving an overall $\chi(\mathbf{R}) = 229.33$. A correlated Gaussian noise process is generated by using the first-order autoregressive model (AR(1)), $v(k+1) = \rho v(k) + v_0(k)$, where $v_0(k)$ is a white Gaussian noise process and ρ is the correlation parameter ($\rho = 0.7$). Simulations were done with the following parameters: For the Fast RI algorithm: $\beta = 0.993$ and $\mu_0 = 0.0009$. For the RLS algorithm: $\beta = 0.993$. For the VSSLMS algorithm: $\gamma = 0.00048$, $\alpha = 0.97$, $\mu_{min} = 0.0047$ and $\mu_{max} = 0.05$. For the DCTLMS algorithm: $\gamma = 0.002$, $\beta = 0.993$, $\epsilon = 8 \times 10^{-4}$, $\mu = 0.02$ and $M = 10$. Fig. 4.6 shows that the Fast RI and RLS algorithms converge to the same MSE almost at the same time, whereas the VSSLMS and DCTLMS algorithms converge to the same MSE but at a much slower rate than the Fast RI and RLS algorithms. Even though the RLS and the fast RI algorithms have similar performances in AWGN and ACGN noise environments, the computational complexity of the Fast RI algorithm is considerably lower as shown in Fig. 3.2 and Table 3.2

4.2.3. Additive White Impulsive Noise

Due to man-made noise [74], underwater acoustic noise [75], atmospheric noise [76], etc, the noise added to the received signal can not be modeled using the Gaussian distribution. This type of noise which has a heavy-tailed distribution is characterized by outliers and may be better modeled using a Gaussian mixture model. In order to study the effects of the impulsive components (outliers) of the noise process in the noise cancellation setting, an impulsive noise process is generated by the probability density function [18]: $f = (1-\epsilon)\, G\left(0, \sigma_n^2\right) + \epsilon G\left(0, \kappa\sigma_n^2\right)$ with vari-

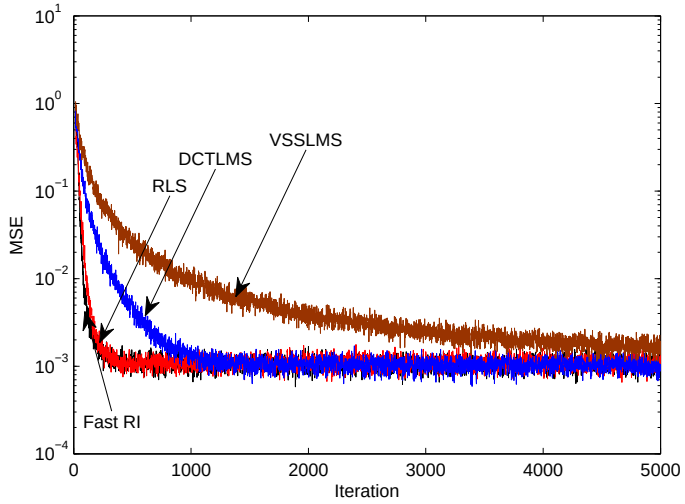

Figure 4.6. The ensemble MSE for the fast RI, RLS, VSSLMS and DCTLMS in ACGN.

ance σ_f^2 given as: $\sigma_f^2 = (1-\epsilon)\,\sigma_n^2 + \epsilon\kappa\sigma_n^2$, where $G\left(0, \sigma_n^2\right)$ is a Gaussian probability density function with zero mean and variance σ_n^2 that represents the nominal background noise. $G\left(0, \kappa\sigma_n^2\right)$ represents the impulsive component of the noise model, where ϵ is the probability and $\kappa \geq 1$ is the strength of the impulsive components, respectively. The signal is assumed to be corrupted with an Additive White Impulsive Noise (AWIN) process. The white impulsive noise process is generated with the parameters: $\epsilon = 0.2$ and $\kappa = 100$. Simulations were done with the following parameters: For the Fast RI algorithm: $\beta = 0.993$ and $\mu_0 = 0.0009$. For the RLS algorithm: $\beta = 0.993$. For the VSSLMS algorithm: $\gamma = 0.00048$, $\alpha = 0.97$, $\mu_{min} = 0.0047$ and $\mu_{max} = 0.05$. For the DCTLMS algorithm: $\gamma = 0.002$, $\beta = 0.993$, $\epsilon = 8 \times 10^{-4}$, $\mu = 0.02$ and $M = 10$. Fig. 4.7 shows that the fast RI and RLS algorithms converge to the same MSE at the same time, whereas the VSSLMS and DCTLMS algorithms converge to the same MSE but at a much slower rate than the Fast RI and RLS algorithms.

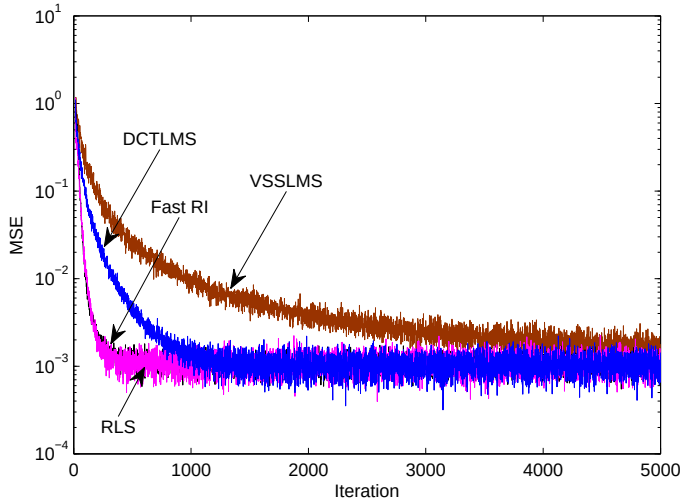

Figure 4.7. The ensemble MSE for the fast RI, RLS, VSSLMS and DCTLMS in AWIN.

4.2.4. Additive Correlated Impulsive Noise

In this part, the signal is assumed to be corrupted with an Additive Correlated Impulsive Noise (ACIN) process. A correlated Gaussian noise process is generated by using the AR(1) model described in section 4.2.2, where $v_0(k)$ here is a white impulsive noise process described in section 4.2.3 with the same parameters. Simulations were done with the following parameters: For the Fast RI algorithm: $\beta = 0.993$ and $\mu_0 = 0.0009$. For the RLS algorithm: $\beta = 0.993$. For the VSSLMS algorithm: $\gamma = 0.00048$, $\alpha = 0.97$, $\mu_{min} = 0.0047$ and $\mu_{max} = 0.05$. For the DCTLMS algorithm: $\gamma = 0.002$, $\beta = 0.993$, $\epsilon = 8 \times 10^{-4}$, $\mu = 0.02$ and $M = 10$. Fig. 4.8 shows that, initially RI is converging slightly faster than the RLS algorithm. This may be due to the better robustness of the RI algorithm, which is the result of using the autocorrelation matrix estimate, unlike the RLS which uses the inverse autocorrelation matrix. The VSSLMS and DCTLMS algorithms converge to the same MSE but at a much slower rate than the other algorithms.

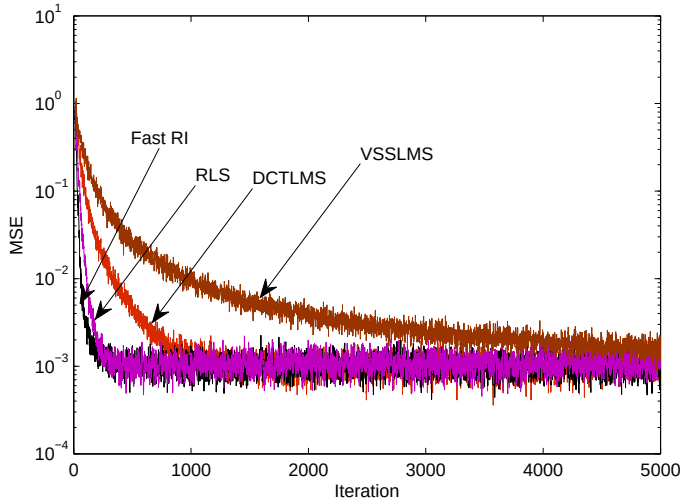

Figure 4.8. The ensemble MSE for the fast RI, RLS, VSSLMS and DCTLMS in ACIN.

4.3. System Identification

In the system identification problem [13], [14] with the block diagram shown in Fig. 4.9, the MSE is considered as $E\{(y(k) - \hat{y}(k))^2\}$. Due to the sensitivity of this MSE to the Toeplitz approximation used in the fast RI algorithm, this algorithm does not perform as good as the RI algorithm in the system identification setting. Hence, the performances of the 2^{nd} order RI and the 1^{st} order RI algorithms are compared with those of the RLS, SFTRLS and TDVSS algorithms. All the algorithms, in this section, were implemented using a filter length of $N = 16$ taps, eigenvalue spread $\chi(\mathbf{R}) = 244.38$ and the simulations were obtained by averaging 100 Monte-Carlo independent runs. The input signal ($x(k)$ in Fig. 4.9) was generated using (4.1). The unknown system coefficients are given in (4.3).

$$h(k) = 10^{-3}[-1 \quad -2 \quad 10 \; 20 \; -35 \; -65 \; 130 \; 450 \; 450$$
$$130 \quad -65 \; -35 \; 20 \; 10 \; -2 \; -1]^T. \tag{4.3}$$

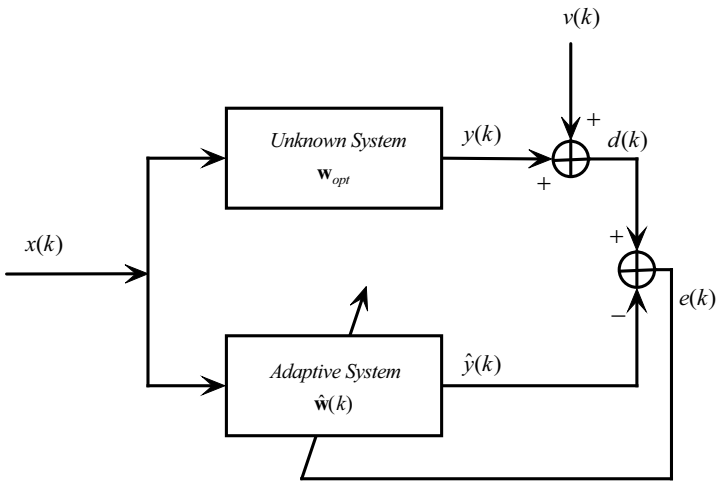

Figure 4.9. Block diagram of system identification.

4.3.1. Additive White Gaussian Noise

To test the performance of the algorithms, the input signal $x(k)$ is assumed to be corrupted with an AWGN process with zero mean and variance ($\sigma_v^2 = 0.000225$). Simulations were done with the following parameters: For the 2^{nd} order RI algorithm, $\beta = 0.997$, $\mu_0 = 0.15$. For the RI algorithm, $\beta = 0.991$, $\mu_0 = 0.00146$. For the RLS algorithm, $\beta = 0.991$. For the SFTRLS algorithm: $\lambda = 0.991$, $\kappa_1 = 1.5$, $\kappa_2 = 2.5$ and $\kappa_3 = 1$, [15]. For the TDVSS algorithm, $\alpha = 0.99$, $\beta = 0.9$, $\epsilon = 0.025$, $\mu_{min} = 0.0047$, $\mu_{max} = 0.05$, $\gamma = 0.001$, $L = 10$. Fig.4.10 (which provides the performance of the algorithms when the unknown system is a lowpass filter) shows that even though the RLS algorithm initially converges faster, the RI and the RLS algorithms converge finally to the same mean-square error (MSE=-50dB) at approximately 400 iterations. Also, the 2^{nd} order RI algorithm converges with them to a much lower MSE (MSE=-59dB), which shows the advantage of the 2^{nd} order RI algorithm over the other algorithms. The SFTRLS and TDVSS algorithms

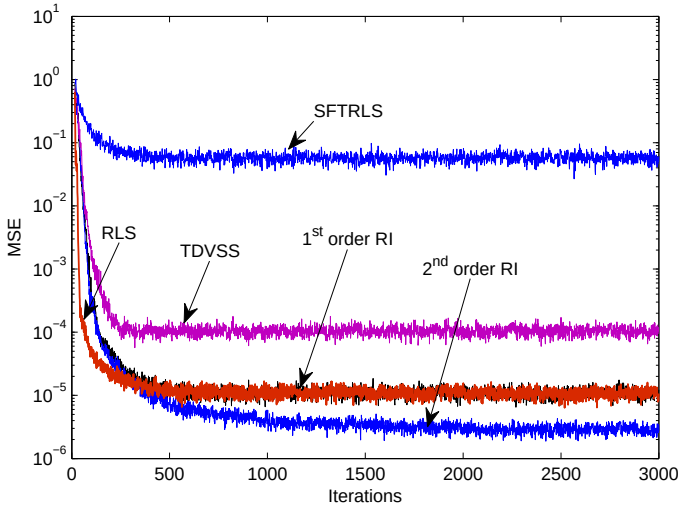

Figure 4.10. The ensemble MSE for 2^{nd} Order RI, RI, RLS, SFTRLS and TDVSS in AWGN.

converge to a relatively higher MSE compared with that of the other algorithms (MSE=-15dB, -40dB, respectively). Another important point here is that it is not possible for the other algorithms to achieve such a low MSE, whatever parameters are chosen. Additionally, it should be noted that the computational complexity of the 2^{nd} order RI algorithm is lower than that of the RLS algorithm and very comparable with that of the RI algorithm.

Fig. 4.11 shows the performance of the algorithms with the same parameters when the unknown system is the bandpass filter given by (4.4). It is noted that the performance of the 1^{st} and 2^{nd} order RI algorithms is slowed down. However, the 1^{st} order RI algorithm converges to a lower MSE (1 dB better) than the RLS algorithm and the 2^{nd} order RI algorithm converges with to a much lower MSE (10dB better) than the RLS algorithm.

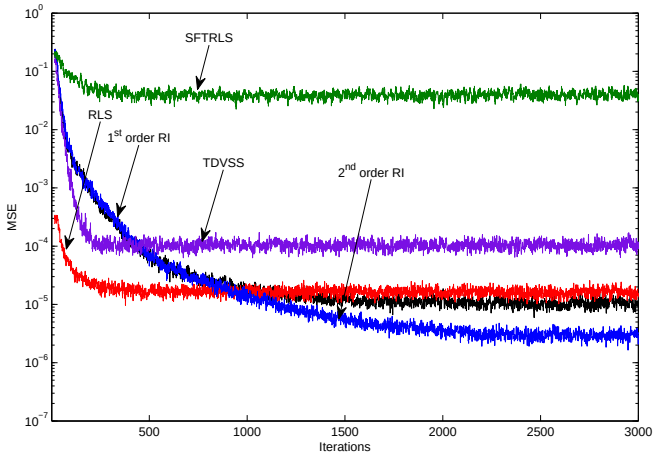

Figure 4.11. The ensemble MSE for 2^{nd} Order RI, RI, RLS, SFTRLS and TDVSS in AWGN ($h_{bp}(k)$ given by (4.4)).

$$h_{bp}(k) = 10^{-4}[-468 \quad -248 \quad 312\ 1085\ 1121 \quad -223 \quad -2804\ 2785\ 2802$$
$$-2226 \quad -1116\ 1080 \quad -315 \quad -244\ 470 \quad -1]^T. \quad (4.4)$$

4.3.2. Additive Correlated Gaussian Noise

In order to test the robustness of the algorithms mentioned above against changes in the environment, the input signal $x(k)$ is assumed to be corrupted by an ACGN. A correlated Gaussian noise process is generated by using the first-order autoregressive model (AR(1)) described in section 4.2.2 with $v(k)$ being a white Gaussian noise process with zero mean and variance ($\sigma_v^2 = 0.000576$). Simulations were done with the following parameters: For the 2^{nd} order RI algorithm, $\beta = 0.997$, $\mu_0 = 0.15$. For the RI algorithm, $\beta = 0.991$, $\mu_0 = 0.00146$. For the RLS algorithm, $\beta = 0.991$. For the SFTRLS algorithm: $\lambda = 0.991$, $\kappa_1 = 1.5$, $\kappa_2 = 2.5$ and $\kappa_3 = 1$. For the TDVSS algorithm, $\alpha = 0.99$, $\beta = 0.9$, $\epsilon = 0.025$, $\mu_{min} = 0.0047$,

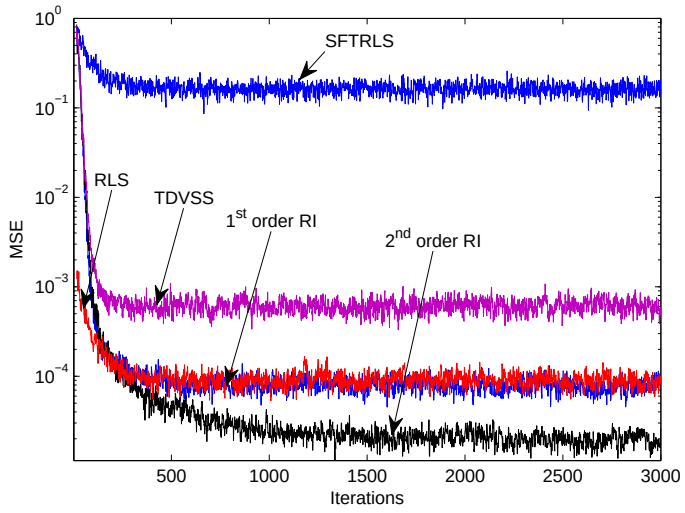

Figure 4.12. The ensemble MSE for 2^{nd} Order RI, RI, RLS, SFTRLS and TDVSS in ACGN.

$\mu_{max} = 0.05$, $\gamma = 0.001$, $L = 10$. Fig.4.12 shows that the RLS algorithm is converging faster at the beginning, but the RI and the RLS algorithms converge finally to the same mean-square error (MSE=-40dB) at approximately 400 iterations. Also, the 2^{nd} order RI algorithm converges to a much lower MSE (MSE=-49dB). This shows that the 2^{nd} order RI algorithm is less affected by the noise type than the other algorithms. The SFTRLS and TDVSS algorithms converge to a higher MSE compared to that of the other algorithms (MSE=-9.5dB, -34dB, respectively).

In order to study the robustness of the proposed algorithm in impulsive noise environments, its performance is compared to those of the conventional RLS and the Robust RLS [17] algorithms in a system identification setting. The unknown system was an Finite-Impulse-Response (FIR) filter which was obtained using the MATLAB commands $\mathbf{h} = \mathrm{fir1}(M-1, \omega_n)$ and $\mathbf{w}_{opt} = \mathbf{h}/\mathrm{norm}(\mathbf{h}, 2)$ with $M = 37$ and $\omega_n = 0.3$. All the algorithms were implemented using a filter length of $N = 37$

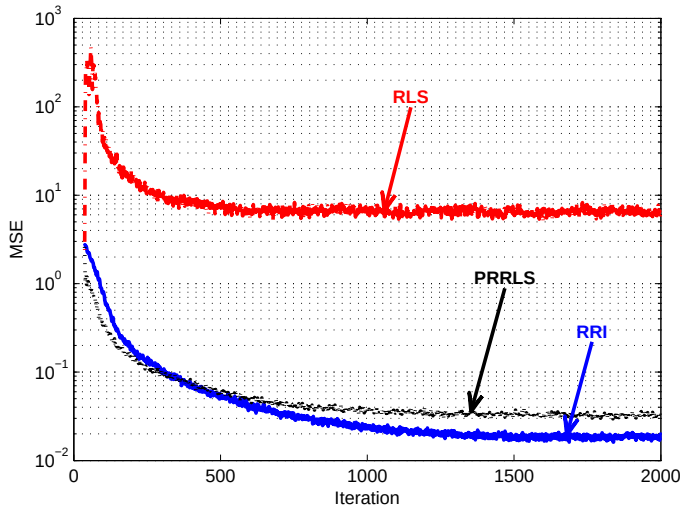

Figure 4.13. The ensemble MSE for the Robust RI, conventional RLS, and Robust RLS algorithms in AWIN ($\epsilon = 0.01$, $\kappa = 10000$).

taps and SNR$=$ 0dB. The input signal was generated using (4.1). All results are obtained by averaging 1000 independent runs.

4.3.3. Additive White Impulsive Noise

In this experiment, an AWIN process is generated by the model described in Section 4.2.3. The white impulsive noise process is generated with $\epsilon = 0.01$, $\kappa = 10000$. Simulations were done with the following parameters: For the proposed RRI algorithm: $\beta = 0.993$, $\mu_0 = 0.006$. For the conventional RLS algorithm: $\beta = 0.993$. For the PRRLS algorithm [17]: $\lambda = 0.993$. Fig. 4.13 shows that the conventional RLS algorithm fails to converge, whereas the Robust RI and Robust RLS algorithms converge approximately at the same time (1400 iterations), but the proposed algorithm converges to a lower MSE (MSE=-19dB for Robust RI and MSE=-17dB for Robust RLS). Fig. 4.14 shows the performance gain of the proposed RRI algorithm relative to that of the PRRLS algorithm in impulsive noise.

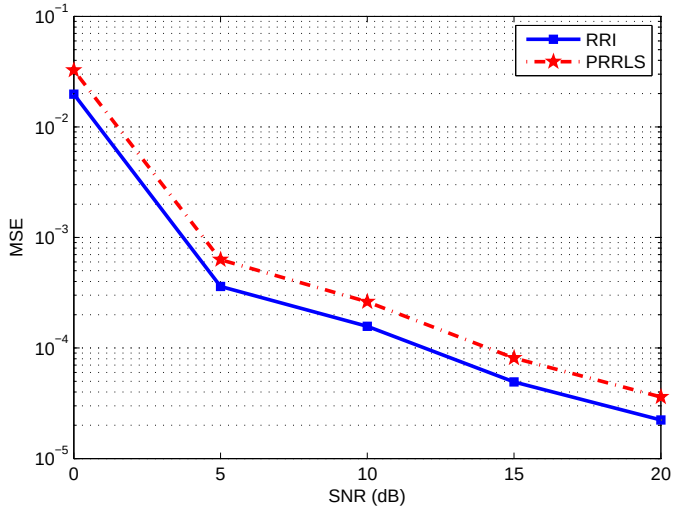

Figure 4.14. The ensemble MSE for the Robust RI and Robust RLS algorithms in AWIN for different SNR's ($\epsilon = 0.01$, $\kappa = 10000$).

On the other hand, in order to observe the effects of the impulsive components (outliers) of the noise process more prominently, a white impulsive noise process is generated with $\epsilon = 0.2$, $\kappa = 100$. These parameters are used to model severely impulsive noise [18]. All other parameters are kept the same as before. Fig. 4.15 shows that the proposed RRI converges to a relatively lower MSE (MSE=-17dB) than the conventional RLS (MSE=3dB) and PRRLS (MSE=-11dB) algorithms. By comparing Fig. 4.13 and Fig. 4.15, it is important to note that the heavier outliers in the noise process has decreased the performance of the PRRLS algorithm significantly (approximately by 7dB). However, the proposed RRI is more robust to impulsive noise with a performance decrease of 2dB only.

4.3.4. Additive Correlated Impulsive Noise

In this experiment, the signal created by (4.1) is assumed to be corrupted with an ACIN process. A correlated impulsive noise process is generated by using the first-order autoregressive model (AR(1)), $v(k+1) = \rho v(k) + n_0(k)$, where $n_0(k)$ is a

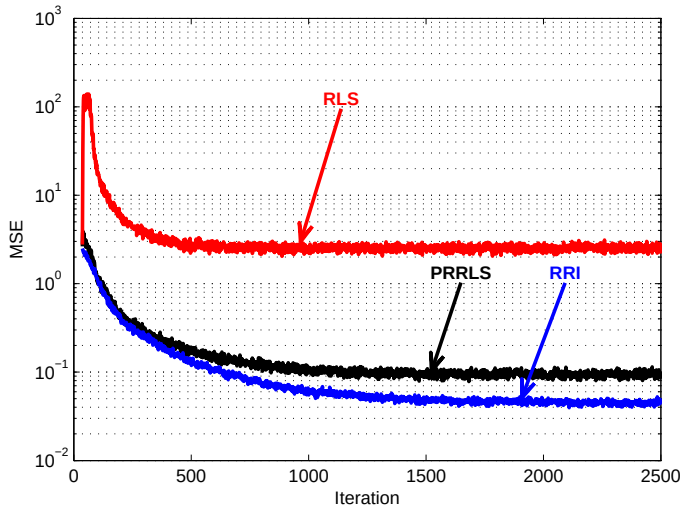

Figure 4.15. The ensemble MSE for the Robust RI, conventional RLS, and Robust RLS algorithms in AWIN ($\epsilon = 0.2$, $\kappa = 100$).

white impulsive noise process created as described in Section 4.2.3 and ρ is the correlation parameter ($\rho = 0.7$). Simulations were done with the following parameters: For the proposed RRI algorithm: $\beta = 0.993$, $\mu_0 = 0.006$. For the conventional RLS algorithm: $\beta = 0.993$. For the PRRLS algorithm [17]: $\lambda = 0.993$. Fig. 4.16 shows that the conventional RLS algorithm again fails to converge. On the other hand, the RRI and PRRLS algorithms both converge at the same time (1500 iterations) with the MSE of the proposed algorithm 1dB lower than that of the PRRLS algorithm.

4.3.5. Nonstationary Additive White Gaussian Noise Environment

In this section, to show the robustness of the proposed algorithms in nonstationary environments, the input signal $x(k)$ given in (4.1) is assumed to be corrupted with AWGN with zero mean and variance ($\sigma_v^2 = 0.000225$). The impulse response given in (4.3) is abruptly corrupted by AWGN with zero mean and variance ($\sigma_n^2 = 0.0025$) after iteration $k = 10^4$.

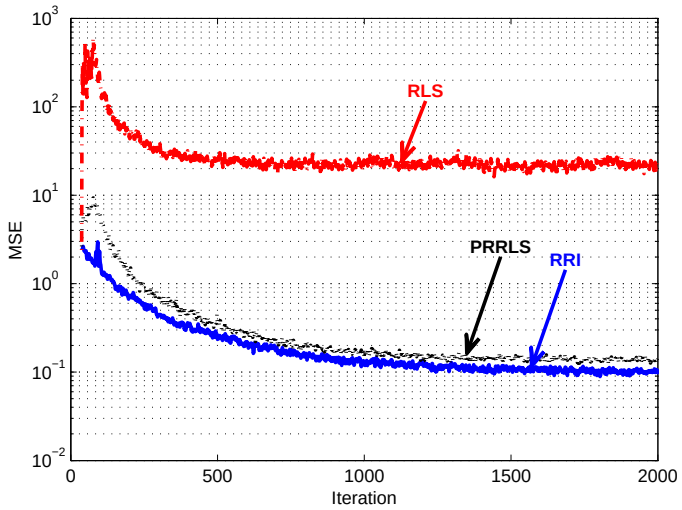

Figure 4.16. The ensemble MSE for the Robust RI, conventional RLS, and Robust RLS algorithms in ACIN ($\epsilon = 0.01$, $\kappa = 10000$).

Simulations were done with the following parameters: For the 2^{nd} order RI algorithm, $\beta = 0.995$, $\mu_0 = 0.18$. For the RI algorithm, $\beta = 0.995$, $\mu_0 = 0.00075$. For the RLS algorithm, $\beta = 0.995$. For LMS, $\mu = 0.0075$ Fig.4.17 shows that both RI algorithms converge at the same time with 1dB lower MSE for 2^{nd} order RI. However, even the RLS algorithm converges slightly faster than RI and 2^{nd} order RI algorithms, but its MSE is worse by 3dB compared to that of the RI and 4dB compared to that of the 2^{nd} order RI algorithm. The LMS algorithm converges much slower than all the others to a higher MSE.

4.4. Channel Equalization

In this section, the channel equalization model of a linear dispersive communication channel is described. The block diagram of the model is depicted in Fig. 4.18. The two random-number generators (1 and 2) in the model are used to generate the transmitted signal x_n and the additive noise at the receiver input, respectively. The sequence x_n is a Bernoulli sequence with $x_n = \pm 1$; the random variable x_n has a

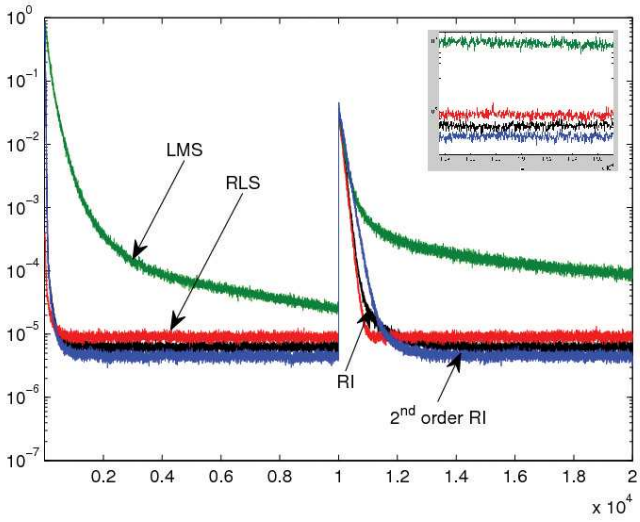

Figure 4.17. The ensemble MSE for 2^{nd} Order RI, RI, RLS and LMS in nonstationary AWGN environment.

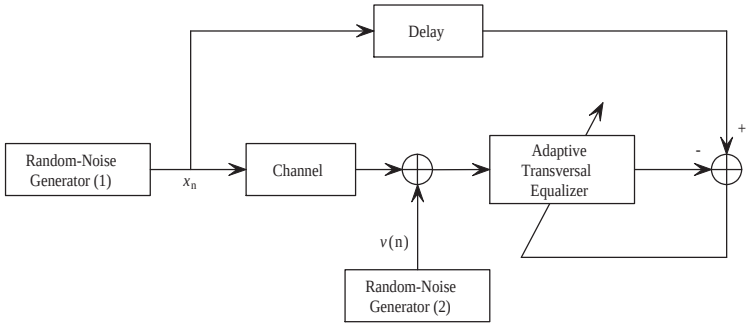

Figure 4.18. Block diagram of the adaptive equalization model.

zero mean and variance 1, and $v(n)$ has a zero mean and variance dependent on the desired SNR. The impulse response of the channel is defined by:

$$h(n)=\begin{cases} \frac{1}{2}\left[1+\cos\left(\frac{2\pi}{W}(n-2)\right)\right], & n=1,2,3, \\ 0, & \text{otherwise} \end{cases} \tag{4.5}$$

where W controls the eigenvalue spread of the autocorrelation matrix. In the simulations of this section, the performance of the Fast RI algorithm is compared to

those of the RLS and SFTRLS algorithms in the channel equalization problem described above. All the experiments were done with the parameters: filter length of $N = 11$ taps, SNR$= 30$dB, $W = 3.3$ and delay of $\Delta = 7$.

4.4.1. Additive White Gaussian Noise

In this experiment, the input signal x_n is assumed to be corrupted with an AWGN process after passing through the channel. Simulations were done with the following parameters: For the Fast RI algorithm: $\beta = 0.985$, $\mu_0 = 0.004$. For the RLS algorithm: $\beta = 0.985$. For the SFTRLS algorithm, [15]: $\lambda = 0.991$, $\kappa_1 = 1.5$, $\kappa_2 = 2.5$ and $\kappa_3 = 1$. Fig. 4.19 shows that the Fast RI and RLS algorithms converge approximately at the same time but the Fast RI algorithm converges to a lower mse (mse$=-28$dB for Fast RI and mse$=-24$dB for RLS), whereas, the SFTRLS algorithm converges to a much higher mse of -9dB. Furthermore, it should be noted that the Fast RI algorithm does not require the inversion of the autocorrelation matrix which will ensure its numerical stability. However, the RLS algorithm may face numerical stability problems due to the loss of Hermitian symmetry and loss of positive definiteness of the inverse autocorrelation matrix, [8].

4.4.2. Additive Correlated Gaussian Noise

The signal is assumed to be corrupted with an ACGN process. A correlated Gaussian noise process is generated by using the model described in Section 4.2.2 with a correlation parameter ($\rho = 0.7$). Simulations were done with the following parameters: For the Fast RI algorithm: $\beta = 0.985$, $\mu_0 = 0.004$. For the RLS algorithm: $\beta = 0.985$. For the SFTRLS algorithm, [15]: $\lambda = 0.991$, $\kappa_1 = 1.5$, $\kappa_2 = 2.5$ and $\kappa_3 = 1$. Fig. 4.20 shows that the Fast RI and RLS algorithms converges approximately at the same time but the Fast RI algorithm converges to a slightly lower mse

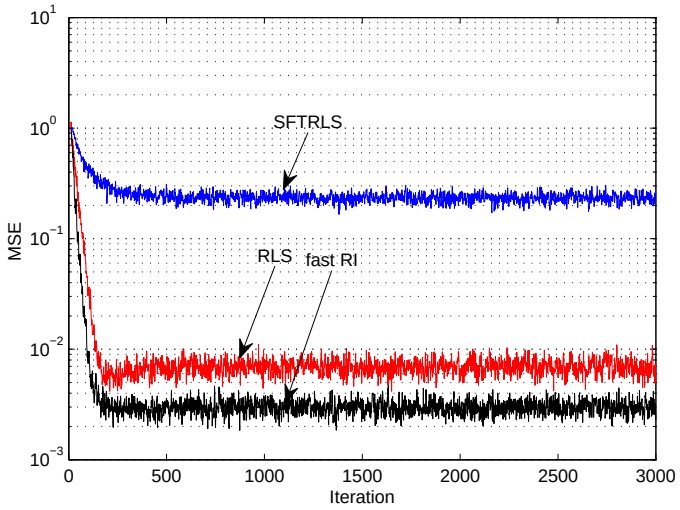

Figure 4.19. The ensemble MSE for Fast RI, RLS and SFTRLS in AWGN.

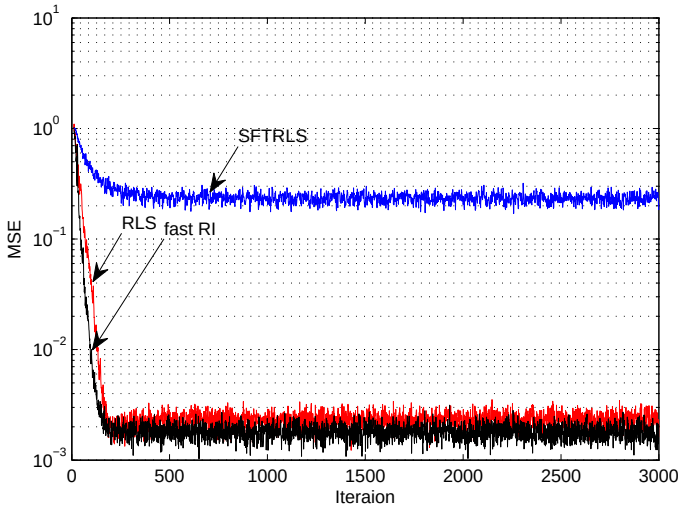

Figure 4.20. The ensemble MSE for Fast RI, RLS and SFTRLS in ACGN.

(mse=-29dB for Fast RI). Even though the SFTRLS algorithm has computational complexity $O(N)$, it converges to a much higher mse of -9dB and slower than the Fast RI and the RLS algorithms. Additionally, the forgetting factors λ in SFTRLS algorithm and β in RLS algorithm, has to to be chosen such that their values are very close to 1. However, this poses a limitation for the algorithms since small values of these parameters may be required in non-stationary environments. On the other hand, in the case of the Fast RI algorithm, there is no restriction on β (In the current experiment, it is selected to be the same as the one in the RLS algorithm). In the case of the SFTRLS algorithm, with $\lambda < 0.991$, the algorithm faces stability problems.

4.4.3. Additive White Impulsive Noise

In this section, an impulsive noise process is generated by the model described in Section 4.2.3. The white impulsive noise process is generated with $\epsilon = 0.2$, $\kappa = 100$. Simulations were done with the following parameters: For the Fast RI algorithm: $\beta = 0.985$, $\mu_0 = 0.004$. For the RLS algorithm: $\beta = 0.985$. For the SFTRLS algorithm: $\lambda = 0.991$, $\kappa_1 = 1.5$, $\kappa_2 = 2.5$ and $\kappa_3 = 1$. Fig. 4.21 shows that the Fast RI and RLS algorithms converges approximately at the same time but the Fast RI algorithm converges to a lower mse (mse=-27dB for Fast RI and mse=-21dB for RLS), whereas, the SFTRLS algorithm converges to a much higher mse of -9dB.

4.4.4. Additive Correlated Impulsive Noise

A correlated impulsive noise process is generated by using the first-order autoregressive model (AR(1)), $v(k+1) = \rho v(k) + v_0(k)$, where $v_0(k)$ is a white impulsive noise process created by the process described in Section 4.2.3 with $\epsilon = 0.2$,

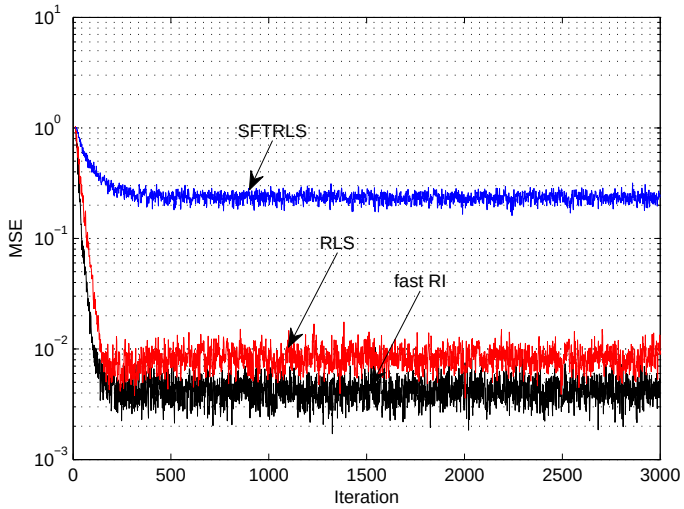

Figure 4.21. The ensemble MSE for Fast RI, RLS and SFTRLS in AWIN.

$\kappa = 100$, and ρ is the correlation parameter ($\rho = 0.7$). Simulations were done with the following parameters: For the Fast RI algorithm: $\beta = 0.985$, $\mu_0 = 0.004$. For the RLS algorithm: $\beta = 0.985$. For the SFTRLS algorithm: $\lambda = 0.991$, $\kappa_1 = 1.5$, $\kappa_2 = 2.5$ and $\kappa_3 = 1$. Fig. 4.22 shows that the Fast RI and RLS algorithms converges approximately at the same time but the Fast RI algorithm converges to a lower mse (mse=-27dB for Fasr RI and mse=-21dB for RLS), whereas, the SFTRLS algorithm converges to a much higher mse of -9dB.

4.5. Acoustic Echo Cancellation

Acoustic Echo Cancellation (AEC) is one of the most important research topics in acoustic signal processing. It is mainly motivated by the increasing demand for hands-free speech communication [21]. A classical AEC scenario is shown in Fig. 4.23. A speech signal $\mathbf{x}(k)$ from the far-end side is broadcasted in an acoustic room by means of a loudspeaker. A microphone is present in the room for recording a local signal $\mathbf{v}(k)$ (near-end speech signal) which is to be transmitted back to the far-end side. An acoustic echo path exists between the loudspeaker and the

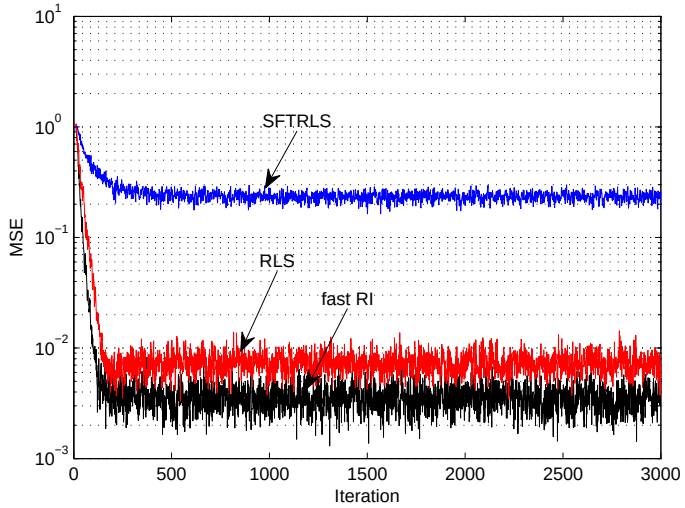

Figure 4.22. The ensemble MSE for Fast RI, RLS and SFTRLS in ACIN.

microphone such that the recorded microphone signal $\mathbf{s}(k) = \mathbf{u}(k) + \mathbf{v}(k)$ contains an undesired echo component $\mathbf{u}(k)$ in addition to the near-end speech signal component $\mathbf{v}(k)$. If the echo path transfer function is modelled as an FIR filter $\mathbf{f}(k)$, then the echo component can be considered as a filtered version of the loudspeaker signal, ($\mathbf{u}(k) = \mathbf{x}(k) * \mathbf{f}(k)$). The main objective in an AEC is to identify the unknown room impulse response $\mathbf{f}(k)$ and hence to subtract an estimate of the echo signal from the microphone signal. In this way, a signal without any echoes, $\mathbf{d}(k) = \mathbf{s}(k) - \mathbf{x}(k) * \mathbf{w}(k)$, is sent to the far-end side, where $\mathbf{w}(k)$ is an estimate of $\mathbf{f}(k)$, [77].

In the simulations, the performance of the RI algorithm is compared to that of the RLS algorithm in an AEC problem [78]. Both algorithms were implemented using a filter length of $N = 100$ taps. The performance is measured in terms of ERLE as given below:

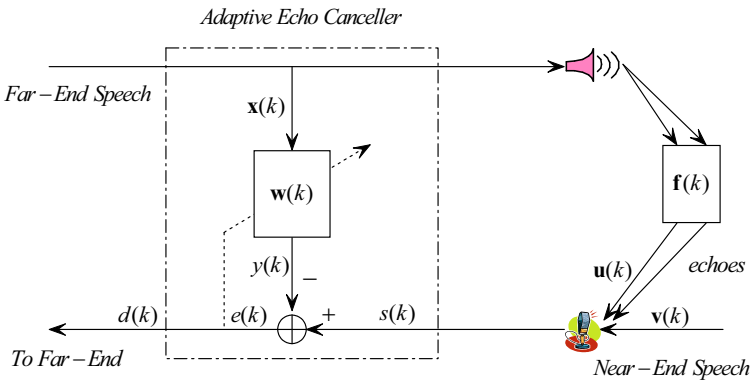

Figure 4.23. Block Diagram of a Typical Acoustic Echo Canceller.

$$ERLE = 10log_{10}\left(\frac{P_d}{P_e}\right). \tag{4.6}$$

where P_d is the power of the difference between the actual speech signal and the predicted one, and P_e is the power of the error signal.

As shown in Fig. 4.23, the far end signal comes to the microphone at near end user in the acoustic room after passing some echo paths forming the unwanted echo signal. These echo paths are assumed to be forming the impulse response, $\mathbf{f}(k)$, in Fig. 4.23. In this experiment, this impulse response is assumed to have the magnitude response shown in Fig. 4.24. Simulations were done with these parameters: For RI algorithm: $\mu_0 = 0.00146$ and $\beta = 0.998$. For RLS algorithm: $\beta = 0.998$. Fig. 4.25 shows that the RLS algorithm is better than the RI algorithm at the beginning in terms of ERLE. After a while the RI algorithm outperforms the RLS algorithm and at the end of iterations the RI outperforms the RLS algorithm by 13dB. Also, It should be noted that the computational complexity of the RI algorithm is much less

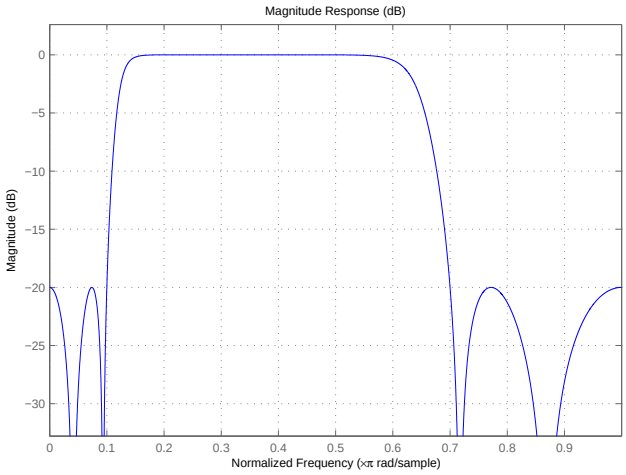

Figure 4.24. The magnitude response of the acoustic room.

than that of the RLS algorithm because the filter length N, here, is relatively high.

4.6. Image Processing Applications

4.6.1. Image Deconvolution

Blind image deconvolution [22], [79]-[81] refers to the problem of reconstructing the original from a degraded observation without knowing either the true image or the degradation process. Mathematically, this can be represented as [22]:

$$g(x,y) = f(x,y) * h(x,y), \tag{4.7}$$

where $*$ stands for the 2D convolution operation, $g(x,y)$ is the blurred image, $f(x,y)$ is the true image and $h(x,y)$ is the Point Spread Function (PSF) given in (4.8).

$$h(x,y) = \begin{pmatrix} -0.035 & -0.65 & -0.35 \\ 0.45 & 0.09 & 0.45 \\ 0.13 & -0.65 & 0.13 \end{pmatrix} \tag{4.8}$$

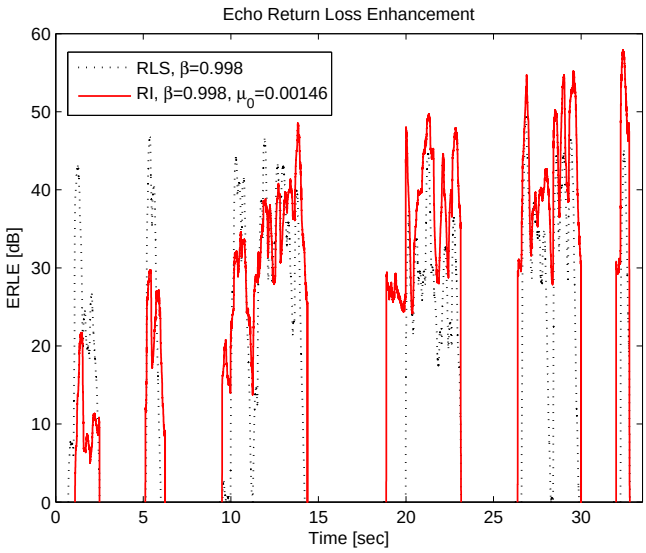

Figure 4.25. The ERLE for RI and RLS.

The performance of the proposed 2D RI algorithm is compared to that of the 2D RLS algorithm in the image deconvolution problem described above. Both algorithms were implemented using a 2D FIR filter of size 3×3 taps (using a 4×4 filter size does not provide any extra significant performance of the algorithms as shown in Table 4.1). For the 2-D RI algorithm: $\mu_0 = 0.000146$, $\beta = 0.999$. For the 2-D RLS algorithm: $\beta = 0.999$.

Fig. 4.26(a) shows the original image 'Lena', Fig. 4.26(b) shows the degraded image which is created by convolving the original image with the PSF in (4.8). Fig. 4.26(c) shows the image restored by the proposed 2D RI algorithm and Fig. 4.26(d) shows the image restored by the 2D RLS algorithm. By inspection, it can be seen that the image restored by the proposed algorithm has sharper edges than that recovered by the 2D RLS algorithm. In terms of PSNR, the resulting image of the proposed algorithm gives 25.64 dB whereas the 2D RLS algorithm provides

25.04 dB. Also, unlike the 2D RLS algorithm, the proposed algorithm does not require the update of the inverse autocorrelation matrix which provides a more stable performance.

Table 4.1. Performance of RI and RLS Algorithms with Different Filter Sizes.

	PSNR (dB)	
filter size / algorithm	3×3	4×4
RI	25.64	25.65
RLS	25.04	25.04

4.6.2. 2D Adaptive Line Enhancer

In this section, the proposed algorithm [82] is applied to a 2D adaptive line enhancer (ALE) in [66]. Fig.4.27 shows the block diagram of the ALE. In Fig.4.27 $\mathbf{x}(n_1, n_2)$ is the original image with noise, $\mathbf{u}(n_1, n_2)$ is the delayed signal of the desired response.

Here, the advantage of the proposed algorithm appears more because the image used (Checker Board) is more correlated than Lena's image. Both algorithms were implemented using a 2D FIR filter of size 3×3 taps. For the 2D RI algorithm: $\mu_0 = 0.000146$, $\beta = 0.995$ for Fig. 4.28(c) and $\beta = 0.9995$ for Fig. 4.28(e). For the 2-D RLS algorithm: $\beta = 0.995$ for Fig. 4.28(d) and $\beta = 0.9995$ for Fig. 4.28(f). The noise is white Gaussian with PSNR$= 12$ dB.

Fig. 4.28(a) shows the original image 'Checker Board', Fig. 4.28(b) shows the image with noise. Fig. 4.28(c) shows the restored image by the proposed 2D RI algorithm with $\beta = 0.995$ and Fig. 4.28(d) shows the restored image by the 2D

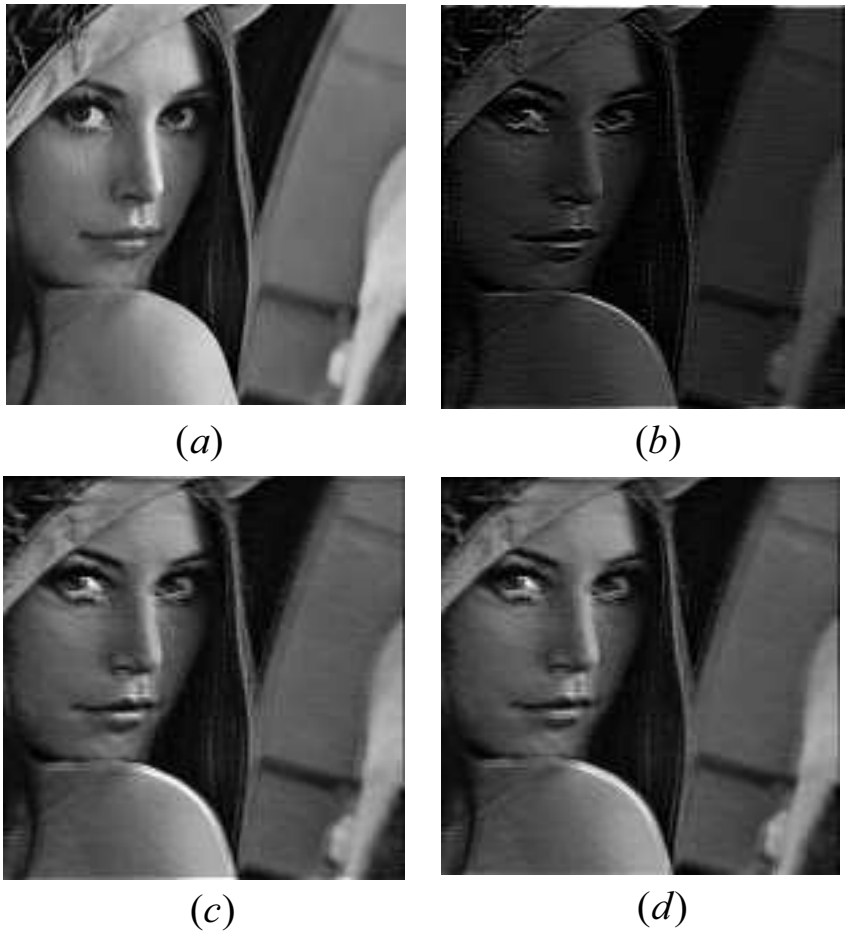

Figure 4.26. (a) Original image, (b) Blurred image, (c) Restored image using the 2D RI algorithm, and (d) Restored image using the 2D RLS algorithm.

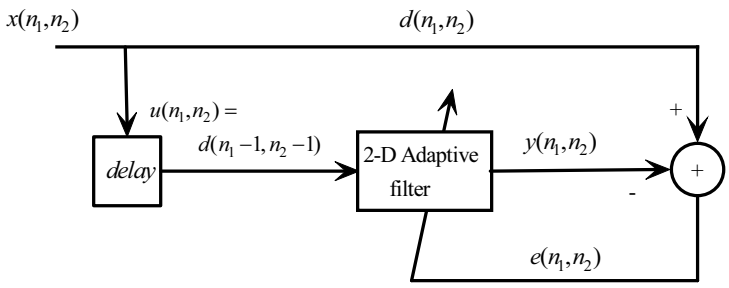

Figure 4.27. The block diagram of a 2D adaptive line enhancer.

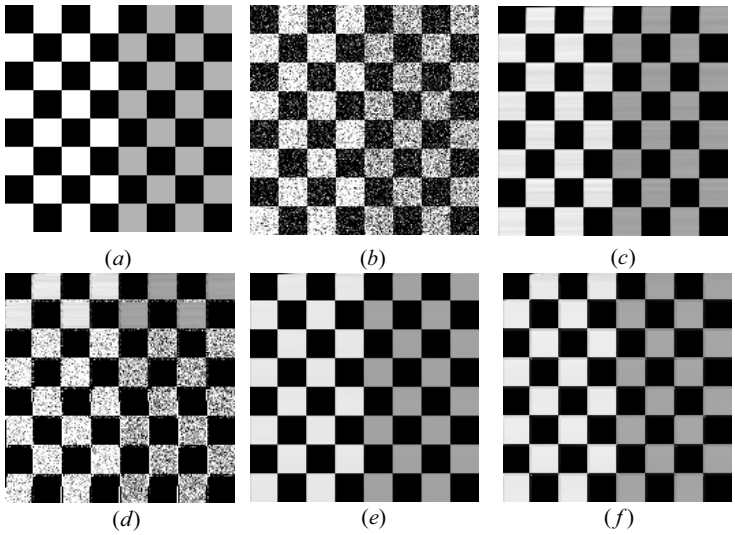

Figure 4.28. (a) Original image, (b) Image with noise, (c) Restored image using the 2D RI ($\beta = 0.995$), (d) Restored image using the 2D RLS ($\beta = 0.995$), (d) Restored image using the 2D RI algorithm ($\beta = 0.9995$), and (f) Restored image using the 2D RLS algorithm ($\beta = 0.9995$).

RLS algorithm with $\beta = 0.995$. We note that the RLS algorithm diverges with relatively low values of β whereas the RI algorithm converges at the same value of β. Fig. 4.28(e) shows the restored image by the proposed 2D RI algorithm with $\beta = 0.9995$ and Fig. 4.28(f) shows the restored image by the 2D RLS algorithm with $\beta = 0.9995$. Even though both algorithms converge, it is clear that the image restored by the proposed algorithm has sharper edges than that recovered by the 2D RLS algorithm even by inspection. For Figs. 4.28(e) and (f), in terms of PSNR the proposed algorithm provides 27.1 dB whereas the 2D RLS algorithm provides 25.4 dB.

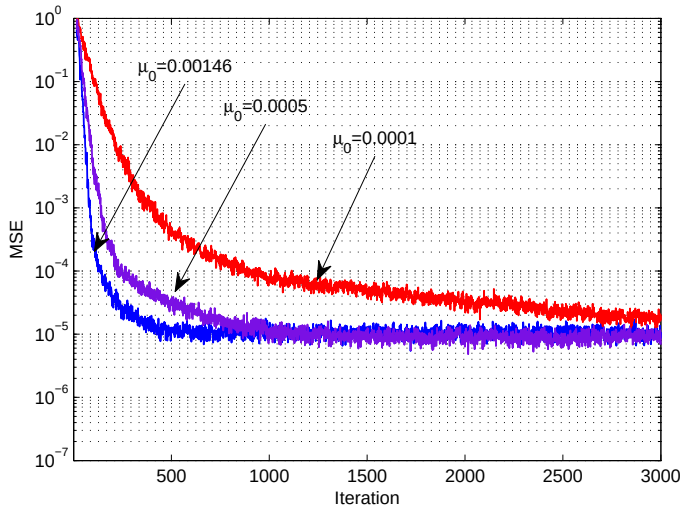

Figure 4.29. The ensemble MSE for the RI algorithm in AWGN with different values of μ_0.

4.7. Effect of the Filter Parameters on the MSE

4.7.1. Effect of the Forgetting Factor, Step-Size and Filter Length on Steady-State MSE

In order to study the effect of changing the filter parameters, μ_0 and β have been changed while all the other parameters of the RI algorithm were kept the same. Fig. 4.29 shows the steady-state MSE with different values of μ_0. Fig. 4.29 shows that reducing the value of μ_0 slows the convergence of the RI algorithm to the steady-state, but the algorithm converges to the same MSE in all the cases, as expected. Secondly, β has been varied while all the other parameters of the RI algorithm were kept the same. Fig. 4.30 shows the steady-state MSE with different values of β. In Fig. 4.30 we note that increasing the value of β leads to a better performance in terms of MSE but to a certain value. After that value, increasing β will not reduce the MSE, but it will result in an unstable performance of the algorithm as shown in Fig. 4.30 at $\beta = 0.9929$.

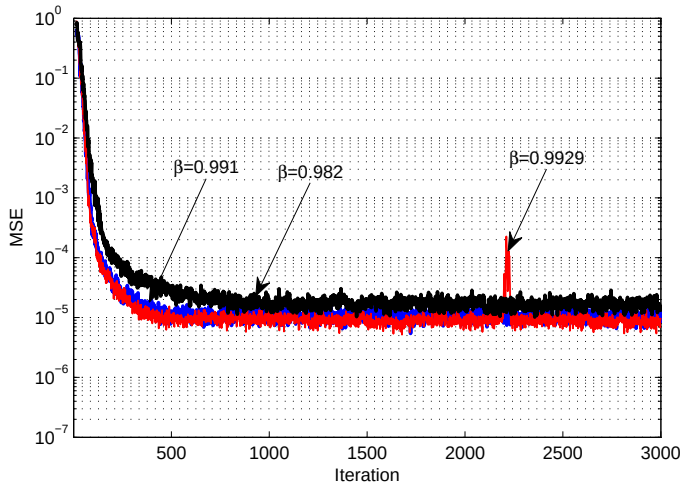

Figure 4.30. The ensemble MSE for the RI algorithm in AWGN with different values of β.

To study the effect of changing the filter length, N was changed while all the other parameters of the RI algorithm were kept the same. Fig. 4.31 shows the steady-state MSE with respect to different tap-lengths. It is clear from Fig. 4.31 that after a certain value of tap-length (In this case $N = 16$), changing the filter length will not affect the steady-state MSE. However, increasing N will increase the number of computations needed for updating the tap-weight vector as shown in Fig. 3.2.

4.7.2. Effect of the Forgetting Factor on the RI Adaptive Algorithm in System Identification

In this section, we simulate the system identification problem shown in Fig. 4.9. The input signal was generated using (4.1) [13]. The system noise $\nu(k)$ is assumed to be AWGN with zero mean and variance ($\sigma_\nu^2 = 0.015$).

Fig. 4.32 shows the normalized residual parameter estimate given in (3.106) of the RI and RLS algorithms for different values of the filter length, N. This term was

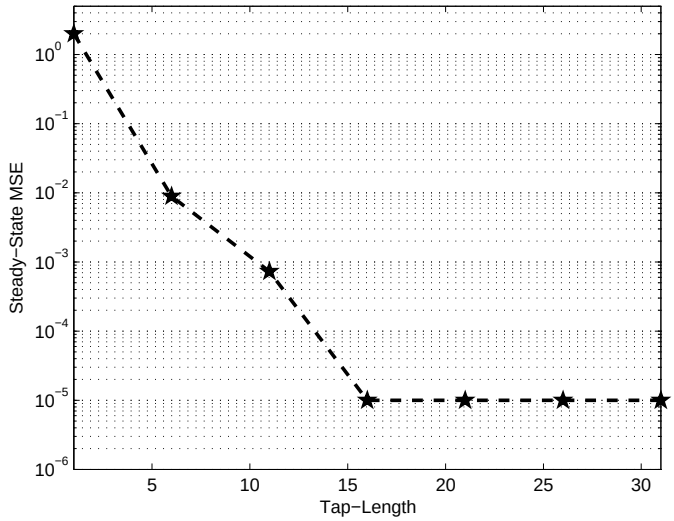

Figure 4.31. The curve of steady-state MSE with respect to the tap-length.

averaged over the last 500 samples of the steady-state of both algorithms. It is seen that, when the filter length is relatively small (8-32 taps) the RI algorithm is less sensitive to the changes in the forgetting factor, β, and the filter length, N, than the RLS algorithm. On the other hand, when the filter length is relatively high (i.e. $N = 128$ taps), the RI algorithm becomes sensitive to the value of β (the larger the N the larger the β should be).

In order to investigate the *leakage* phenomenon, two separate experiments were done. In order to study the effect of the forgetting factor β on the leakage phenomenon, the filter length is held constant ($N = 32$) and the forgetting factor, β, is assigned different values (i.e. $\beta = 0.9, 0.99, 0.999$). The recovered signal ($e(k)$) and the residual error ($e_r(k) = e(k) - \nu(k)$) were plotted for each value of β for both, the RI and the RLS, algorithms. It can be seen from Fig. 4.33 and Fig. 4.34 that the *leakage* in the RI algorithm case is much less than that of the RLS algorithm, especially, when the forgetting factor β is relatively small (i.e $\beta = 0.9, 0.99$).

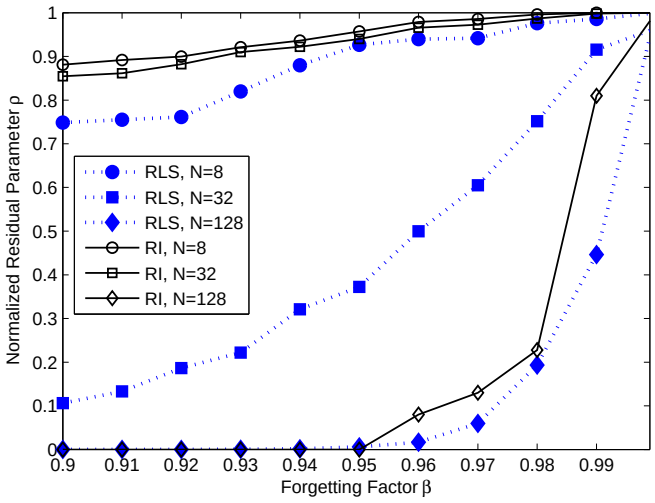

Figure 4.32. Normalized residual parameter ρ for different values of the forgetting factor β and the filter length N.

Also, to observe this *leakage* in the frequency domain, we repeat the previous experiments and plot the power spectral density (PSD) of both, $e(k)$ and $e_r(k)$ in Fig. 4.35 and Fig. 4.36. The conclusions are, almost, the same as those in the time domain; the recovered signal is strongly attenuated when the value of β is relatively small for the RLS algorithm. However, in the case of the RI algorithm, this attenuation become much less especially when the forgetting factor and/or the filter length are relatively small.

In order to study the effect of the filter length on the *leakage* phenomenon, now, the forgetting factor, β, is held constant ($\beta = 0.99$) and the filter length is assigned different values (i.e. $N = 8, 32, 64$ taps). The recovered signal ($e(k)$) and the residual error ($e_r(k)$) are plotted for each value of N for the RI and the RLS algorithms. Fig. 4.37 and Fig. 4.38 show that the *leakage* in the RI algorithm case is much less than that of the RLS algorithm, especially, when the filter length N is relatively small

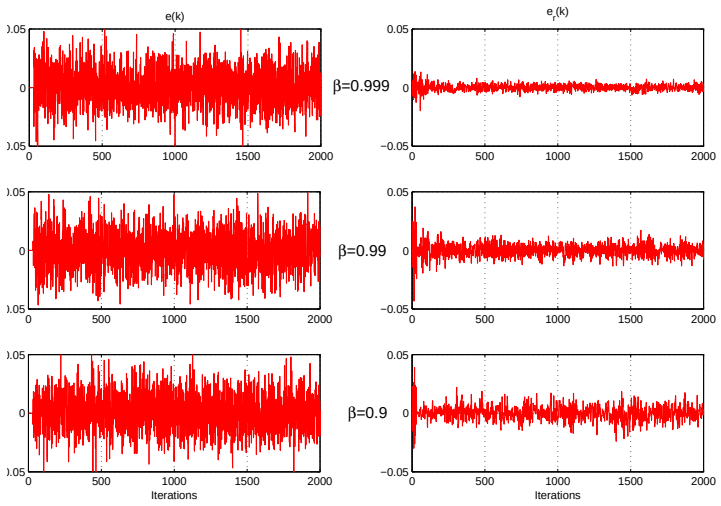

Figure 4.33. The leakage phenomenon of The RI algorithm in time domain $N = 32$ and different values of β.

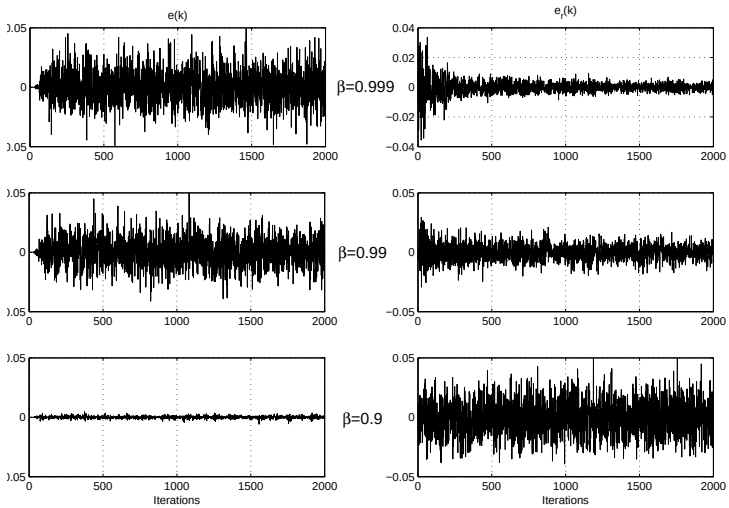

Figure 4.34. The leakage phenomenon of The RLS algorithm in time domain $N = 32$ and different values of β.

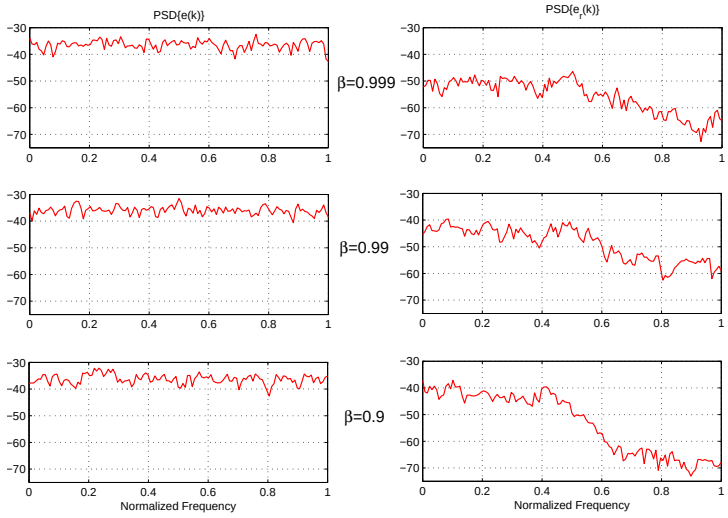

Figure 4.35. Power Spectral Density of Fig. 4.33.

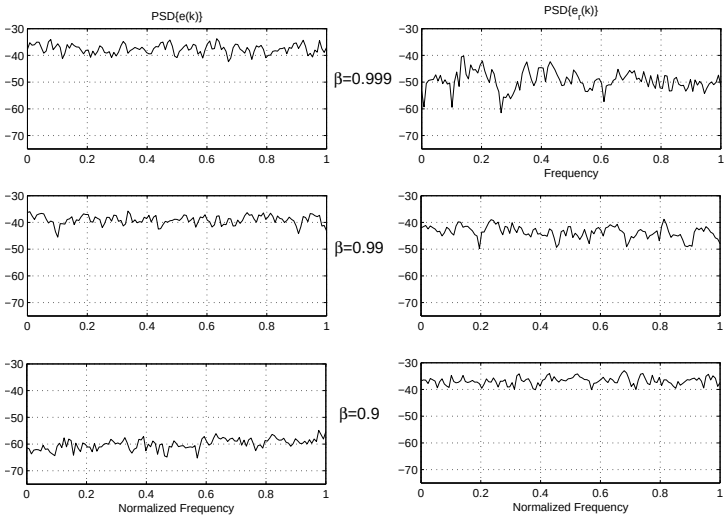

Figure 4.36. Power Spectral Density of Fig. 4.34.

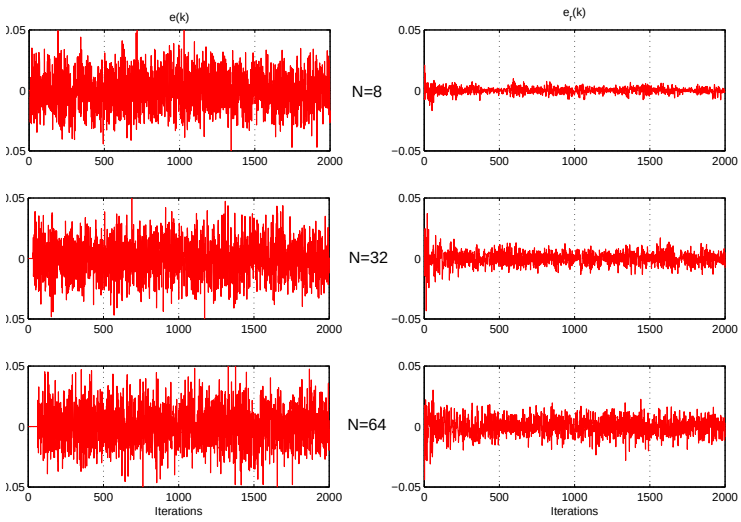

Figure 4.37. The leakage phenomenon of the RI algorithm in time domain

$\beta = 0.99$ and different values of N.

(i.e $N = 8, 32$ taps).

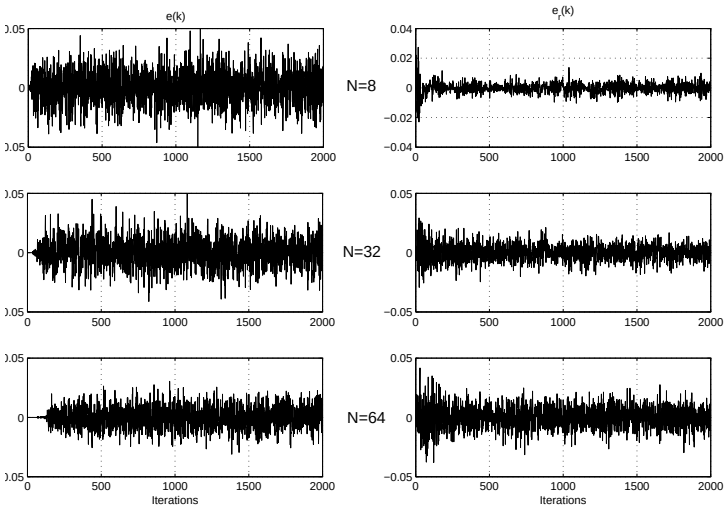

Figure 4.38. The leakage phenomenon of the RLS algorithm in time domain

$\beta = 0.99$ and different values of N.

Chapter 5

CONCLUSIONS AND FUTURE WORK

5.1. Conclusions

In this work, a new FIR RI adaptive filtering algorithm is introduced. This algorithm has been proposed to overcome some of the difficulties experienced with the above-mentioned adaptive filters. The approach uses a variable step-size and the instantaneous value of the autocorrelation matrix in the coefficient update equation that leads to an improved performance. Convergence analysis of the algorithm has been presented. The ensemble-average learning curve of the RI algorithm is derived and compared with those of the RLS and LMS algorithms. A general fast implementation technique, which significantly reduces the computational complexity, of the RI algorithm is presented. A robust version of the RI algorithm, which leads to an improved performance in impulsive noise environments is presented. The effect of the forgetting factor on the performance of the RI algorithm is derived. Also, a 2D version of the RI algorithm is introduced. Finally, a second-order version of the RI algorithm, which provides further improvement in the performance, is derived. Following are some of the conclusions which are drawn from the work carried out:

- It is shown that the RI and the fast RI algorithms outperform the conventional adaptive filtering algorithms in stationary AWGN, ACGN, AWIN and ACIN environments of noise cancellation and system identification settings. Also,

it is shown that in a non-stationary AWGN environment of a system identification setting, both RI algorithms converge at the same time with a slightly better MSE for the 2^{nd} order RI. However, even the RLS algorithm converges slightly faster than RI and 2^{nd} order RI algorithms, but its MSE is worse than those of the RI and 2^{nd} order RI algorithms. The LMS algorithm converges much slower than all the others to a higher MSE. With the improved version, the second-order RI, it is shown that at the expense of a slight increase in the computational burden, it becomes possible to attain low MSE levels that do not seem possible with the other algorithms. This is made possible by using the autocorrelation matrix in the update equation of the RI algorithm directly, and not its inverse. On the other hand, using a second order updating of the correlations in the RLS algorithm seems almost impossible. This feature of the RI algorithm may be considered to be a significant contribution to the adaptive filtering field.

- For impulsive noise environments, a robust RI adaptive filtering algorithm that outperforms the robust RLS algorithm in impulsive noise has been proposed. The proposed algorithm employs the L_1 norm of the gain factor of the cross-correlation vector to achieve robust performance. Simulation results show that the proposed algorithm is robust against white and correlated impulsive noise and provides better performance compared to those of the conventional and robust RLS algorithms.

- In system identification setting, the existence of a leakage phenomenon is an obstacle in some adaptive algorithms. We theoretically proved that this leakage is proportional to the system noise, and highly dependent on the values

of the forgetting factor and the filter length. Simulation results validate the theoretical results, and they show that for a relatively small forgetting factor and/or a relatively large filter length, this leakage is high or total in some cases. However, simulations show that the RI algorithm is much less sensitive to these parameters (β and N) than the RLS algorithm. Both the theoretical and experimental results lead to the conclusion that the leakage phenomenon can be avoided when the value of β is very close to unity, in both algorithms.

- Also, the performance of the RI algorithm is compared to that of the RLS algorithm in the AEC problem. Under the same conditions, the RI algorithm has a performance better than RLS in terms of ERLE with a considerable reduction in computational complexity.

- Finally, simulations show that the proposed 2D RI algorithm leads to an improved performance over that of the 2D RLS algorithm in an image deconvolution problem. In an ALE problem, the advantage of the 2D RI is more pronounced. After both algorithms converge, simulation results show that the proposed 2D RI algorithm leads to an improved performance over that of the 2D RLS algorithm.

5.2. Future Work

The RI algorithm has shown superior performance compared to a large group of adaptive algorithm in AWGN, ACGN, AWIN, ACIN stationary and non-stationary environments in noise cancellation and system identification settings. However, future work may include:

- Although the RI algorithm uses a variable step-size, an initial step-size μ_0

should be selected carefully. A better way of selecting this initial step-size may be investigated.

- Also, performance analysis of the RI algorithm in non-stationary environments may be derived.

- Finally, performance of the RI algorithm in communication fading channels may be investigated.

APPENDIX A

.1. Matrix Inversion Lemma

The Matrix Inversion Lemma is given by

$$\left(\mathbf{A}-\mathbf{B}\mathbf{D}^{-1}\mathbf{C}\right)^{-1}=\mathbf{A}^{-1}+\mathbf{A}^{-1}\mathbf{B}\left(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)^{-1}\mathbf{C}\mathbf{A}^{-1}, \tag{1}$$

To prove (1), let us construct the augmented matrix $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$ and its inverse,

$$\begin{pmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{C} & \mathbf{D}\end{pmatrix}^{-1}=\begin{pmatrix}\mathbf{E} & \mathbf{F}\\ \mathbf{G} & \mathbf{H}\end{pmatrix}, \tag{2}$$

now, construct the two products

$$\begin{pmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{C} & \mathbf{D}\end{pmatrix}\cdot\begin{pmatrix}\mathbf{E} & \mathbf{F}\\ \mathbf{G} & \mathbf{H}\end{pmatrix}=\begin{pmatrix}\mathbf{AE}+\mathbf{BG} & \mathbf{AF}+\mathbf{BH}\\ \mathbf{CE}+\mathbf{DG} & \mathbf{CF}+\mathbf{DH}\end{pmatrix}=\begin{pmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{pmatrix}, \tag{3}$$

and

$$\begin{pmatrix}\mathbf{E} & \mathbf{F}\\ \mathbf{G} & \mathbf{H}\end{pmatrix}\cdot\begin{pmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{C} & \mathbf{D}\end{pmatrix}=\begin{pmatrix}\mathbf{EA}+\mathbf{FC} & \mathbf{EB}+\mathbf{FD}\\ \mathbf{GA}+\mathbf{HC} & \mathbf{GB}+\mathbf{HD}\end{pmatrix}=\begin{pmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}\end{pmatrix}, \tag{4}$$

Submatrices in (refeqA3) and (refeqA3) are broken out to form eight matrix equations:

$$\mathbf{AE}+\mathbf{BG}=\mathbf{I} \tag{5}$$

$$\mathbf{AF}+\mathbf{BH}=\mathbf{0} \tag{6}$$

$$\mathbf{CE}+\mathbf{DG}=\mathbf{0} \tag{7}$$

$$\mathbf{CF}+\mathbf{DH}=\mathbf{I} \tag{8}$$

$$\mathbf{EA} + \mathbf{FC} = \mathbf{I} \tag{9}$$

$$\mathbf{EB} + \mathbf{FD} = \mathbf{0} \tag{10}$$

$$\mathbf{GA} + \mathbf{HC} = \mathbf{0} \tag{11}$$

$$\mathbf{GB} + \mathbf{HD} = \mathbf{I} \tag{12}$$

Combining (5) through (12) in various orders produces two sets of equations for **E**, **F**, **G**, and **H** from **A**, **B**, **C**, and **D**:

$$\mathbf{E} = \left(\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C}\right)^{-1} \tag{13}$$

$$\mathbf{F} = -\left(\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C}\right)^{-1}\mathbf{BD}^{-1} \tag{14}$$

$$\mathbf{G} = -\mathbf{D}^{-1}\mathbf{C}\left(\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C}\right)^{-1} \tag{15}$$

$$\mathbf{H} = \mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{C}\left(\mathbf{A} - \mathbf{BD}^{-1}\mathbf{C}\right)^{-1}\mathbf{BD}^{-1} \tag{16}$$

$$\mathbf{H} = \left(\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B}\right)^{-1} \tag{17}$$

$$\mathbf{G} = -\left(\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B}\right)^{-1}\mathbf{CA}^{-1} \tag{18}$$

$$\mathbf{F} = -\mathbf{A}^{-1}\mathbf{B}\left(\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B}\right)^{-1} \tag{19}$$

$$\mathbf{E} = \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}\left(\mathbf{D} - \mathbf{CA}^{-1}\mathbf{B}\right)^{-1}\mathbf{CA}^{-1} \tag{20}$$

The proof is completed by combining either (13) and (17) or (16) and (20). Conditions are that all the involved inverses exist.

.2. Toeplitz Matrix Properties

We say that a square matrix is Toeplitz if:

1. All the elements on its main diagonal are equal.
2. The elements on any other diagonal parallel to the main diagonal are also equal.

It is important to recognize that the Toeplitz property of the autocorrelation matrix **R** is a direct consequence of the assumption that the discrete-time stochastic process represented by the observation vector $\mathbf{x}(k)$ is wide-sense stationary. Indeed, we may state that if the discrete-time stochastic process is wide-sense stationary, then its autocorrelation matrix **R** must be Toeplitz; and, conversely, if the autocorrelation matrix **R** is Toeplitz, then the discrete-time stochastic process must be wide-sense stationary.

.3. Time-Domain Solution of State Equations

In general, a discrete-time system can be represented by the state equations

$$\mathbf{x}[k+1] = \mathbf{A}\mathbf{x}[k] + \mathbf{B}\mathbf{f}[k] \tag{21}$$

$$\mathbf{y}[k] = \mathbf{C}\mathbf{x}[k] + \mathbf{D}\mathbf{f}[k] \tag{22}$$

From (21) it follows that

$$\mathbf{x}[k] = \mathbf{A}\mathbf{x}[k-1] + \mathbf{B}\mathbf{f}[k-1] \tag{23}$$

and

$$\mathbf{x}[k-1] = \mathbf{A}\mathbf{x}[k-2] + \mathbf{B}\mathbf{f}[k-2] \tag{24}$$

$$\mathbf{x}[k-2] = \mathbf{A}\mathbf{x}[k-3] + \mathbf{B}\mathbf{f}[k-3] \tag{25}$$

$$\tag{26}$$

Substituting (24) in (23), we obtain

$$\mathbf{x}[k] = \mathbf{A}^2\mathbf{x}[k-2] + \mathbf{A}\mathbf{B}\mathbf{f}[k-2] + \mathbf{B}\mathbf{f}[k-1] \tag{27}$$

Substituting (25) in (27), we obtain

$$\mathbf{x}[k] = \mathbf{A}^3\mathbf{x}[k-3] + \mathbf{A}^2\mathbf{B}\mathbf{f}[k-3] + \mathbf{A}\mathbf{B}\mathbf{f}[k-2] + \mathbf{B}\mathbf{f}[k-1] \tag{28}$$

Continuing this way (knowing that $\mathbf{x}[1] = \mathbf{A}\mathbf{x}[0] + \mathbf{B}\mathbf{f}[0]$), we obtain

$$\begin{aligned} \mathbf{x}[k] &= \mathbf{A}^k\mathbf{x}[0] + \mathbf{A}^{k-1}\mathbf{B}\mathbf{f}[0] + \mathbf{A}^{k-2}\mathbf{B}\mathbf{f}[1] + \mathbf{B}\mathbf{f}[k-1] \\ &= \mathbf{A}^k\mathbf{x}[0] + \sum_{j=0}^{k-1} \mathbf{A}^{k-1-j}\mathbf{B}\mathbf{f}[j] \end{aligned} \tag{29}$$

This is the desired solution. The first term on the right-hand side represents $x(k)$ when the input $f(k) = 0$. Hence, it is the zero-input component. The second term, by a similar argument, is seen to be the zero-state component.

By using the same procedure, we obtain

$$\mathbf{y}[k] = \mathbf{C}\mathbf{A}^k\mathbf{x}[0] + \sum_{j=0}^{k-1} \mathbf{C}\mathbf{A}^{k-1-j}\mathbf{B}\mathbf{f}[j] + \mathbf{D}\mathbf{f}$$

REFERENCES

[1] S. Haykin, *Adaptive Filter Theory*, 4^{th} ed., Prentice Hall, Upper Saddle River, NJ, 2002.

[2] B. F. Boroujeny, *Adaptive Filters: Theory and Applications*, John Wiley, Baffins Lane, Chichester, 1998.

[3] B. Widrow and E. Walach, *Adaptive Inverse Control, Reissue Edition: A Signal Processing Approach*, Wiley-IEEE Press, 2007.

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

[5] D. Starer and A. Nehorai, "Newton algorithms for conditional and unconditional maximum likelihood estimation of the parameters of exponential signals in noise," *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. 40, No. 6, 1992, pp. 1528-1534.

[6] M. G. Bellanger, *Adaptive Digital Filters*, 2^{nd} ed., Marcel Dekker, New York, 2001.

[7] G. Tummarello, F. Nardini and F. Piazza, "Stepsize control in NLMS acoustic echo cancellation using a neural network approach," *International Symposium on Circuits and Systems*, Vol. 5, May 2003, pp. 705-708.

[8] A. H. Sayed, *Adaptive Filters*, John Wiley & Sons, NJ, 2008.

[9] S. Haykin, A. H. Sayed, J. R. Zeidler, P. Yee and P. C. Wei, "Adaptive tracking of linear time-variant systems by extended RLS algorithms," *IEEE Transactions on Signal Processings*, vol. 45, no. 5, May 1997, pp. 1118 - 1128.

[10] B. Widrow and M. E. Hoff, "Adaptive switching circuits," *IRE WESCON Convention Record*, vol. 4, 1960, pp. 96-104.

[11] R. H. Kwong and E. W. Johnston, "A Variable step-size LMS algorithm," *IEEE Transactions on Signal Processing*, vol. 40, no. 7, July 1992, pp. 1633-1642.

[12] D. I. Kim and P. De Wilde, "Performance analysis of the DCT-LMS adaptive filtering algorithm," *Signal Processing*, vol. 80, no. 8, August 2000, pp. 1629-1654.

[13] R. C. Bilcu, P. Kuosmanen and K. Egiazarian, "A Transform domain LMS adaptive filter with variable step-size," *IEEE Signal Processing Letters*, vol. 9, no. 2, February 2002, pp. 51-53.

[14] R. C. Bilcu, *On Adaptive Least Mean Square FIR Filters: New Implementations and Applications*, Ph.D. Dissertations, Tempere University of Technology, Finland, 2004.

[15] P. S. R. Diniz, *Adaptive Filtering Algorithms and Practical Implementation*, 3^{rd} Ed., Springer, LLC, NY, 2008.

[16] M. M. Chansarkar and U. B. Desai, "A Robust recursive least squares algorithm," *IEEE Transactions on Signal Processings*, vol. 45, no. 7, July 1997, pp. 1726-1735.

[17] M.Z.A. Bhotto and A. Antoniou, "Robust recursive least-squares adaptive-filtering algorithm for impulsive-noise environments," *IEEE Signal Processing Letters*, vol. 18, no. 3, 2011, pp. 185-188.

[18] H. Delic and A. Hocanin, "Robust detection in DS/CDMA," *IEEE Transactions on Vehicular Technology*, vol. 51, no. 1, January 2002, pp. 155-170.

[19] A. Sugiyama, R. Miyahara and K. Masanori;, "An adaptive noise canceller with adaptive delay compensation for a distant noise source," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 265-268.

[20] C. Y. Chi, C. C. Feng, C. H. Chen and C. Y. Chen, *Blind Equalization and System Identification: Batch Processing Algorithms, Performance and Applications*, Springer, Germany, 2006.

[21] S.L. Gay and J. Benesty, *Acoustic Signal Processing for Telecommunication*, Kluwer Academic Publishers, Norwell, 2000.

[22] D. Kundur and D. Hatzinakos, "A Novel blind deconvolution scheme for image restoration using recursive filtering," *IEEE Transactions on Signal Processing*, vol. 26, no. 2, 1998, pp. 375-390.

[23] S. Haykin and T. K. Bhattacharya, "Modular learning strategy for signal detection in a nonstationary environment," *IEEE Transactions on Signal Processing*, vol. 45, no. 6, 1997, pp. 1619-1637.

[24] C. V. Sinn and J. Gotze, "Comparative study of techniques to compute FIR filter weights in adaptive channel equalization," *IEEE International Conference*

on Acoustic, Speech and Signal Processing (ICASSP03), vol. 6, April 2003, pp. 217-220.

[25] X. Guan, X. Chen; and G. Wu, "QX-LMS adaptive FIR filters for system identification," 2^{nd} *International Congress on Image and Signal Processing (CISP2009)*, 2009, pp. 1-5.

[26] B. Allen and M. Ghavami, *Adaptive Array Systems: Fundamentals and Applications*, John Wiley & Sons Ltd, West Sussex, England, 2005.

[27] J. I. Nagumo and A. Noda, "A Learning method for system identification," *IEEE Transactions on Automation and Control*, vol. AC-12, 1967, pp. 282-287.

[28] A. E. Albert and L. S. Gardner, *Stochastic Approximation and Nonlinear Regression*, MIT Press, Cambridge, MA, 1967.

[29] R. R. Bitmead and B. D. O. Anderson, "Lyapunov techniques for the exponential stability of linear difference equations with random coefficients," *IEEE Transactions on Automation and Control*, vol. AC-25, 1980, pp. 782-787.

[30] R. R. Bitmead and B. D. O. Anderson, "Performance of adaptive estimation algorithms in independent random environments," *IEEE Transactions on Automation and Control*, vol. AC-25, 1980, pp. 788-794.

[31] D.I. Pazaitis and A. G. Constantinides, "A Novel kurtosis driven variable step-size adaptive algorithm," *IEEE Transactions on Signal Processing*, vol. 47, no. 3, 1999, pp. 864-872.

[32] C. Paleologu, J. Benesty, S. Ciochina and C. Vladeanu, "Practical variable step-size adaptive algorithms for echo cancellation," *Proceedings of the* 5^{th}

Conference on Speech Technology and Human-Computer Dialogue (SPED 2009), 2009, pp. 1-6.

[33] S. S. Narayan, A. M. Peterson and M. J. Narasimha,"Transform domain LMS algorithm," *IEEE Transactions on Acoustic, Speech and Signal Processing*, vol. ASSP-31, no. 3, June 1983, pp. 609-615.

[34] F. Beaufays, "Transform-domain adaptive filters: an analytical approach," *IEEE Transactions on Signal Processing*, vol. 43, no. 2, 1995, pp. 422-431.

[35] R. Hastings-James, R. and M. W. Sage, "Recursive generalised-least-squares procedure for online identification of process parameters," *Proceedings of the Institution of Electrical Engineers*, vol. 116, no. 12, 1969, pp. 2057-2062.

[36] V. Panuska, "An adaptive recursive-least-squares identification algorithm," 8^{th} *IEEE Symposium on Adaptive Processes and Decision and Control*, vol. 8, part 1, 1969, pp. 65-69.

[37] G. O. Glentis, K. Berberidis and S. Theodoridis, "Efficient least squares adaptive algorithms for FIR transversal filtering," *IEEE Signal Processing Magazine*, July 1999, pp. 13-41,

[38] G. O. Glentis, "Efficient fast recursive least-squares adaptive complex filter using real valued arithmetic," *Elsevier Signal Processing*, vol. 85, 2005, pp. 1759-1779.

[39] S. Makino and Y. Kaneda, "A New RLS algorithm based on the variation characteristics of a room impulse response," *IEEE International Conference on*

Acoustics, Speech, and Signal Processing (ICASSP-94), vol. 3, 1994, pp. 373-376.

[40] K. Maouche and D. T. M. Slock, "Fast subsampled-updating stabilized fast transversal filter (FSU SFTF) RLS algorithm for adaptive filtering," *IEEE Transactions on Signal Processing*, vol. 48, no. 8, 2000, pp. 2248-2257.

[41] E. M. Eksioglu and A. K. Tanc, "RLS algorithm with convex regularization," *IEEE Signal Processing Letters*, vol. 18, no. 8, 2011, pp. 470-473.

[42] J. Wang, "A variable forgetting factor RLS adaptive filtering algorithm," 3^{rd} *IEEE International Symposium on Microwave, Antenna, Propagation and EMC Technologies for Wireless Communications*, 2009, pp. 1127-1130.

[43] S. Ciochina, C. Paleologu, J. Benesty and A. A. Enescu, "On the influence of the forgetting factor of the RLS adaptive filter in system identification," *International Symposium on Signals, Circuits and Systems* (ISSCS 2009), 2009, pp. 1-4.

[44] D. T. M. Slock and T. Kailath, "Numerically stable fast transversal filters for recursive least squares adaptive filtering," *IEEE Transactions on Signal Processing*, vol. 39, January 1991, pp. 92-113.

[45] A. Benallal and A. Gilloire, "A New method to stabilize fast RLS algorithms based on a first-order of the propagation of numerical errors," *IEEE International Conference on Acoustics, Speech, and Signal Processing* (ICASSP-88), vol. 3, 1988, pp. 1373-1376.

[46] S. Haykin, A. H. Sayed, J. R. Zeidler, P. Yee and P. C. Wei, "Adaptive tracking of linear time-variant systems by extended RLS algorithms," *IEEE Transactions on Signal Processing*, vol. 45, no. 5, May 1997, pp. 1118 - 1128.

[47] B. Seyfe and S. Valaee, "A New Choice of penalty function for robust multiuser detection based on M-estimation," *IEEE Transactions on Communications Theory*, vol. 53, 2005, pp. 224-227.

[48] Y. Zou and S. C. Chan, "A Robust quasi-Newton adaptive filtering algorithm for impulse noise suppression," *IEEE International Symposium on Circuits and Systems*, May 2001, pp. 677-680.

[49] Y. Zou, S. C. Chan and T. S. Ng, " Robust M-estimate adaptive filtering," *IEE Proceedings - Vision, Image and Signal Processing*, vol. 148, no. 4, 2001, pp. 289-294.

[50] M. S. Ahmad, O. Kukrer and A. Hocanin, "Recursive inverse adaptive filtering algorithm," *Elsevier Digital Signal Processing*, vol. 21, no. 4, 2011, pp. 491-496.

[51] M. S. Ahmad, O. Kukrer and A. Hocanin, "An efficient recursive inverse adaptive filtering algorithm for channel equalization," *European Wireless Conference* (EWC 2010), Lucca, Italy, April 2010, pp. 88-92.

[52] M. S. Ahmad, O. Kukrer and A. Hocanin, "Recursive inverse adaptive filtering algorithm," *Fifth International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control* (ICSCCW 2009), 2009, pp. 1-3.

[53] M. S. Ahmad, O. Kukrer and A. Hocanin, "Analysis of the recursive inverse adaptive algorithm and an improved second-order version," *IEEE Transactions of Signal Processing*, 2011, (Under Revision).

[54] S. Qiao, "Fast adaptive RLS algorithms: a generalized inverse approach and analysis," *IEEE Transactions on Signal Processings*, vol. 39, no. 6, , June 1991, pp. 1455 - 1459.

[55] L. Shu-Hung and C. F. So, "Gradient-based variable forgetting factor RLS algorithm in time-varying environments," *IEEE Transactions on Signal Processings*, vol. 53, no. 8, , 2005, pp. 3141 - 3150.

[56] G. V. Moustakides, "Performance of the forgetting factor RLS during the transient phase," *IEEE Digital Signal Processing Workshop Proceedings* (DSPWS 1996), 1996, pp. 370 - 373.

[57] P. S. R. Diniz, M. L. R. de Campos and A. Antoniou, "Analysis of LMS-Newton adaptive filtering algorithms with variable convergence factor," *IEEE Transactions on Signal Processings*, vol. 43, no. 3, 1995, pp. 617 - 627.

[58] Y. Zhou, S. C. Chan and K. L. Ho, "A New block-exact fast LMS/Newton adaptive filtering algorithm," *IEEE Transactions on Signal Processings*, vol. 54, no. 1, 2006, pp. 374 - 380.

[59] P. P. Mavridis and G. V. Moustakides, "Simplified Newton-type adaptive estimation algorithms," *IEEE Transactions on Signal Processings*, vol. 44, no. 8, 1996, pp. 1932 - 1940.

[60] M. L. R. de Campos and A. Antoniou, "A New quasi-Newton adaptive filtering algorithm," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 44, no. 11, 1997, pp. 924 - 934.

[61] S. Haykin, *Kalman Filtering and Neural Networks*, John Wiley & Sons, NY, 2001.

[62] D. J. Tylavsky and G. R. L. Sohie, "Generalization of the matrix inversion lemma", *Proceedings of the IEEE*, vol. 74, no. 7, pp. 1050-1052, July. 1986.

[63] M. S. Ahmad, O. Kukrer and A. Hocanin, "Robust recursive inverse adaptive algorithm in impulsive noise ," *Circuits, Systems and Signal Processing*, 2011, DOI: 10.1007/s00034-011-9341-6.

[64] C. Paleologu, J. Benesty, and S. Ciochina, "A Robust variable forgetting factor recursive least-squares algorithm for system identification," *IEEE Signal Processing Letters*, vol. 15, 2008, pp. 597-600.

[65] M. S. Ahmad, O. Kukrer, A. Hocanin, "The effect of the forgetting factor on the RI adaptive algorithm in system identification," *International Symposium on Signals, Circuits and Systems* (ISSCS 2011), Romania, June 2011, (Accepted).

[66] M. M. Hadhoud and D. W. Thomas, "The two-dimensional adaptive LMS (TDLMS) algorithm," *IEEE Transactions on Circuits Systems*, vol. CAS-35, no. 5, 1988, pp. 485-494.

[67] W. K. Jenkins and R. P. Faust, "A Constrained two-dimensional adaptive digital filter with reduced computational complexity," *Proceedings of IEEE International symposium on Circuits and Systems*, 1988, pp. 2655-2658.

[68] M. Ohki and S. Hashiguchi, "Two-dimensional LMS adaptive filters," *Technical digest of Korea-Japan Joint Symposium on Information Display*, October 1990, pp. 46-51.

[69] M. Muneyasu, E. Uemoto and T. Hinamoto, "A Novel 2-D adaptive filter based on the 1-D RLS algorithm," *Proceedings of IEEE International Symposium on Circuits and Systems*, vol. 4, 1997, pp. 2317-2320.

[70] J. Sanubari and K. Tokuda, "RLS-type two-dimensional adaptive filter with a *t*-distribution assumption," *Journal of Signal Processing*, vol. 80, no. 12, 2000, pp. 2483-2495.

[71] F. PirahanSiahi, S. N. H. S. Abdullah and S. Sahran, "Adaptive image segmentation based on peak signal-to-noise ratio for a license plate recognition system," *International Conference on Computer Applications and Industrial Electronics* (ICCAIE 2010), 2010, pp. 468-472.

[72] R.A. Soni and W.K. Jenkins, "Projection algorithms for two-dimensional adaptive filtering applications," *Conference Record of the Thirty-First Asilomar Conference on Signals, Systems & Computers*, vol. 1, 1997, pp. 333-337.

[73] M. S. Ahmad, O. Kukrer, A. Hocanin, "Recursive inverse adaptive filter with second order estimation of autocorrelation matrix," *IEEE International Symposium on Signal Processing and Information Technology* (ISSPIT 2010), Luxor, Egypt, 2010, pp. 482-484.

[74] A. Wagstaff and N. Merricks, "Man-made noise measurement programme," *IEE Proceedings- Communications*, vol. 152, no. 3, 2005, pp. 371-377.

[75] J. Ribeiro-Fonseca and L. Correia, "Identification of underwater acoustic noise," *Proceedings of Oceans Engineering for Today's Technology and Tomorrow's Preservation* (OCEANS 1994), vol. 2, 1994, pp. 597-602.

[76] R. K. Potter, "High-frequency atmospheric noise," *Proceedings of the Institute of Radio Engineers*, vol. 19, no. 10, 1931, pp. 1731-1765.

[77] M. Rages and K. C. Ho, "Limits on echo return loss enhancement on a voice coded speech signal," *The* 45^{th} *Midwest Symposium on Circuits and Systems* (MWSCAS-2002), vol. 2,August 2002, pp. 152-155.

[78] M. S. Ahmad, O. Kukrer, A. Hocanin,"Recursive inverse adaptive filtering algorithm in acoustic echo cancellation," 6^{th} *International Symposium on Electrical & Electronic Engineering and Computer Systems* (EEECS 2010), Lefke, North Cyprus, November 2010.

[79] C. Vural and W. A. Sethares, "Blind deconvolution of noisy blurred images via dispersion minimization," 14^{th} *International Conference on Digital Signal Processing* (ICDSP 2002), vol. 2, 2002, pp. 783-786.

[80] C. Vural and W. A. Sethares, "Recursive blind image deconvolution via dispersion minimization," 14^{th} *International Conference on Digital Signal Processing* (ICDSP 2002), vol. 2, 2002, pp. 787-790.

[81] A. Giannoula, "Classification-based adaptive filtering for multiframe blind image restoration," *IEEE Transactions on Image Processing*, vol. 20, no. 2, 2011, pp. 382-390.

[82] M. S. Ahmad, O. Kukrer, A. Hocanin, “A 2-D recursive inverse adaptive algorithm”, *Journal of Signal, Image and Video Processing*, March 2011, DOI: 10.1007/s11760-011-0218-8.

[83] H. J. Kushner, *Approximation and Weak Convergence Methods for Random Processes with Applications to Stochastic System Theory*, MIT Press, Cambridge, MA, 1984.