Improved Robust Adaptive-Filtering Algorithms

by

Md. Zulfiquar Ali Bhotto
B.Sc. Eng, Rajshahi University of Engineering and Technology, Bangladesh, 2002

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctor of Philosophy

in the Department of Electrical and Computer Engineering

Improved Robust Adaptive-Filtering Algorithms

by

Md. Zulfiquar Ali Bhotto
B.Sc. Eng, Rajshahi University of Engineering and Technology, Bangladesh, 2002

ABSTRACT

New adaptive-filtering algorithms, also known as adaptation algorithms, are proposed. The new algorithms can be broadly classified into two categories, namely, steepest-descent and Newton-type adaptation algorithms. Several new methods have been used to bring about improvements regarding the speed of convergence, steady-state misalignment, robustness with respect to impulsive noise, re-adaptation capability, and computational load of the proposed algorithms.

In chapters 2, 3, and 8, several adaptation algorithms are developed that belong to the steepest-descent family. The algorithms of chapters 2 and 3 use two error bounds with the aim of reducing the computational load, achieving robust performance with respect to impulsive noise, good tracking capability and significantly reduced steady-state misalignment. The error bounds can be either prespecified or estimated using an update formula that incorporates a modified variance estimator. Analyses pertaining to the steady-state mean-square error (MSE) of some of these algorithms are also presented. The algorithms in chapter 8 use a so-called *iterative/shrinkage method* to obtain a variable step size by which improved convergence characteristics can be achieved compared to those in other state-of-the-art competing algorithms.

Several adaptation algorithms that belong to the Newton family are developed in chapters 4-6 with the aim of achieving robust performance with respect to impulsive noise, reduced steady-state misalignment, and good tracking capability without compromising the initial speed of convergence. The algorithm in chapter 4 imposes a bound on the L_1 norm of the gain vector in the crosscorrelation update formula to achieve robust performance with respect to impulsive noise in stationary environments. In addition to that, a variable forgetting factor is also used to achieve good tracking performance for applications in nonstationary environments. The algorithm in chapter 5 is developed to achieve a reduced steady-state misalignment and improved convergence speed and a reduced computational load. The algorithm in chapter 6 is essentially an extension of the algorithm in chapter 5 designed to achieve robust performance with respect to impulsive noise and reduced computational load. Analyses concerning the asymptotic stability and steady-state MSE of these algorithms are also presented.

An algorithm that minimizes Reny's entropy of the error signal is developed in chapter 7 with the aim of achieving faster convergence and reduced steady-state misalignment compared to those in other algorithms of this family.

Simulation results are presented that demonstrate the superior convergence characteristics of the proposed algorithms with respect to state-of-the-art competing algorithms of the same family in network-echo cancelation, acoustic-echo cancelation, system-identification, interference-cancelation, time-series prediction, and time-series filtering applications. In addition, simulation results concerning system-identification applications are also used to verify the accuracy of the MSE analyses presented.

Contents

List of Abbreviations

AP	affine projection
BIBO	bounded-input bounded-output
BIDR-LMS	binormalized data reusing LMS
CAP	constrained affine projection
CLMS	constrained LMS
CNLMS	constrained normalized LMS
CRSMAP	constrained robust set-membership affine projection
DS-CDMA	direct-sequence code division multiple access
EMSE	excess mean-square error
FIR	finite-duration impulse response
IIR	infinite-duration impulse response
IP	information potential
ISI	inter-symbol interference
KQN	known quasi-Newton
KRQN	known robust quasi-Newton
LC	linearly constrained
LCMV	linearly constrained minimum variance
LMS	least-mean squares
LS	least-squares
MEE	minimum error entropy
MOE	mean-output error
MSE	mean-squared error
MSD	mean-squared deviation
NLMS	normalized least-mean square
NMEE	normalized minimum error entropy
NPNLMS	nonparametric normalized least-mean squares
NRLS	nonlinear recursive least-squares
PCSMAP	proposed constrained set-membership AP
PNMEE	proposed normalized minimum error entropy
PQN	improved quasi-Newton
PRQN	improved robust quasi-Newton
PRRLS	proposed robust recursive least-squares
PRSMAP	proposed robust set-membership AP

RLM	recursive least-M estimate
RLS	recursive least-squares
RRLS	robust recursive least-squares
RSMAP	robust set-membership affine projection
SHAP	shrinkage affine projection
SHLMS	shrinkage least-mean square
SHNLMS	shrinkage normalized least-mean square
SM	set membership
SMAP	set-membership affine projection
SMBIDR-LMS	set-membership binormalized data reusing least-mean square
SMNLMS	set-membership normalized least-mean square
SNR	signal-to-noise ratio
SSMAP	simplified set-membership affine projection
VLMS	variable step size least-mean square
VMEE	variable step size minimum error entropy
VSSAP	variable step size affine projection

List of Tables

List of Figures

Acknowledgments

First and foremost I would like to thank God for giving me the opportunity to work in the excellent Department of Electrical and Computer Engineering of a very diverse university located at the heart of the most beautiful place on earth, Victoria, BC, Canada.

I also thank my supervisor Professor Andreas Antoniou for giving me the opportunity to work with him and for his patience, sheer guidance, and unconditional support during the four years of my graduate studies. It is a great pleasure for me to have been his coauthor.

Throughout the course work of my Ph.D. program, I was very much fortunate to learn from well known authors at the University of Victoria in signal processing. Many thanks to Drs. Andreas Antoniou, Wu-Sheng Lu, Dale Olesky, T. Aaron Gullivar, Micheal Adams, and Micheal McGuire as what I learned from their classes in my course work is truly invaluable. I thank them all for their high standard of teaching which I truly enjoyed and I have learned a lot from their deep knowledge, discipline, and intuition. I would also like to thank the members of my supervisory committee for their time and kind effort for reviewing the dissertation. I thank Dr. Paulo S. R. Diniz, who brought to my attention adaptive filtering.

I also thank UVic and the Natural Sciences and Engineering Research Council of Canada for their generous financial support of my graduate education during my PhD program. I also take this opportunity to thank the staff of the Department of Electrical and Computer Engineering for their support especially Erik Laxdal, Marry-Anne Teo, Moneca Bracken, Steve Campbell, Lynne Barret, and Vicky Smith.

I would like to express my deep gratitude and appreciation to my undergraduate school, Rajshahi University of Engineering and Technology, Bangladesh, for the outstanding education it gave to me. It was truly invaluable. I will never forget my friends, teachers, and students at Rajshahi University of Engineering and Technology who encouraged me to come to Canada for higher education. It is partly because of their encouragement I have come to UVic. Thanks to all of you.

My experience at UVic is one of the best experiences in my life. This is in large part due to the great friends I met here: Mohammad Fayed, Sabbir Ahmad, Mohamed Yasein, Diego Sorrentino, Akshay Rathore, Sriram Jhala, Clive Antoni, Hythem Elmigri, Yousry Abdel-Hamid, Ahmad Morgan, Jeevan Pant, and Parameswaran Ramachandran. I thank them all for their support and encouragement that helped me to get rid of anxieties and apathy on my academic work. Their friendship will remain a precious treasure in my life.

I thank my father Md. Bodrul Islam and brothers Md. Nakibul Islam, Md. Nasimul Islam, Md. Mahibul Islam, and my sisters Mst. Nasima Khatun, Mst. Nasrin Khatun, and Mst. Nazma Khatun for their friendship, support, and encouragement. Their encouragement and contributions to my life at UVic and before are too numerous to delineate here. I would also like to thank Md. Aminul Islam, Md. Sazzad Hossein, Md. Abu Nur Sharif Chandan, Mst. Hasiba Khatun, Mst. Hasna Khatun, Mst. Shanaz Khatun. Special thanks to my parents-in-law Shahina Sultana and Mohammad Fariduddin for their prayers, love, and good wishes. I would also like to take this opportunity to thank my beloved wife Sonea Ferdosh. Although you came when I already went a long way towards my PhD, had it not been for your prayers, love, and support, completion of my PhD would have taken more time. Last, but not least, my greatest thanks go to my mother Mst. Nurjahan Begum whose sacrifices, blessing, and good wishes have made my journey possible to this stage. Thank you Ma.

Dedication

The most important person in my life, my mother.

Chapter 1

Introduction

Adaptive filters are filters that tune their parameters as time advances to adapt their performance according to some prespecified criterion. Adaptive filters that use the mean square-error (MSE) criterion are the most popular in practice as the mathematical complexities involved are relatively easy to handle. The adaptation algorithms involved start with an initial guess that is based on available information about the system and then refine the guess in successive iterations such that each refinement would improve the solution and eventually converge to the optimal Wiener solution in some statistical sense [1, 2, 3]. Adaptive filters have a wide range of applications including echo cancelation, equalization, noise cancelation, signal prediction, interference suppression, and beamforming [2, 3]. The performance of adaptive filters is evaluated based one or more of the following factors [3]:

1. Convergence speed: This is a measure of the capability of the adaptive filter to achieve fast convergence. It is proportional to the inverse of the number of iterations required to yield a solution that is close enough to the Wiener solution.

2. Excess MSE: This quantity is defined as the amount by which the steady-state misalignment of the adaptive filter exceeds the minimum MSE produced by the optimal Wiener filter.

3. Computational complexity: This is defined as the amount of computation required by the adaptive filter in one iteration which can be measured in terms of the number of arithmetic operations or the CPU time required.

4. Computational load: This is the product of the computational complexity and the number of iterations required by the adaptive filter to converge.

5. Numerical robustness: An adaptive filter is said to be numerically robust when its implementation using finite-word-length arithmetic remains stable indefinitely even for ill-conditioned input signals.

Other measures of adaptive filters are robustness with respect to impulsive noise and tracking capability. In applications where continuous operation is required, the performance of the adaptive filter should not deteriorate due to perturbations brought about by impulsive noise and also it should have good tracking.

Ideally, one would like to have an adaptive filter that is computationally efficient, numerically robust with the highest convergence speed, and also yields the lowest possible excess MSE. In addition, it should be easy to implement the adaptive filter with low-cost, low-precision VLSI chips. In adaptive filters, as in any engineering design problem, it is not possible to achieve all the desirable features simultaneously and, in effect, trade-offs exist. For example, least-mean-squares (LMS) adaptive filters are computationally simple and numerically robust but they have the drawback of very slow convergence, especially when the input signal is colored [2, 3]. On the other hand, recursive-least-squares (RLS) adaptive filters have fast convergence and reduced excess MSE but they are computational complex and, in addition, they are subject to serious numerical problems [2, 3]. These are the two families of algorithms often used to illustrate the two extremes of the existing trade-off spectrum for the design of adaptive filters. Over the years, steady research has been going on towards improving the performance of adaptive filters with respect to different performance measures. Our work is also aimed in that direction.

1.1 The State-of-the-Art

The general set up of an adaptive filter is illustrated in Fig. 1.1 where $\boldsymbol{x}_k \in \mathcal{R}^{M\times 1}$ denotes the input signal, $y_k = \boldsymbol{x}_k^T \boldsymbol{w}_{k-1} \in \mathcal{R}^{1\times 1}$ is the adaptive-filter output, $d_k \in \mathcal{R}^{1\times 1}$ defines the desired signal, and k is the iteration number. The *a priori* error signal is computed as $e_k = d_k - y_k$ and then used to form a time-varying objective function which is optimized online with respect to the filter weights by using a suitable algorithm commonly referred to as an *adaptation algorithm*. The objective functions used in adaptive filters approximate the unknown mean-squared error (MSE) function

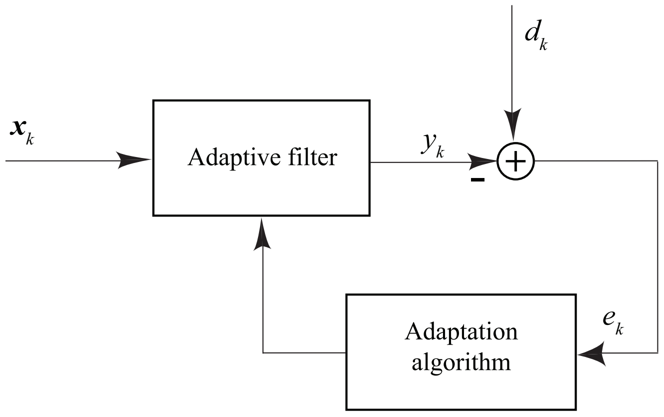

Figure 1.1: Basic adaptive-filter configuration.

$E[e_k^2]$ in various ways so that the adaptation algorithm yields a solution that is close enough to the optimal Wiener solution. In this section, several classes of adaptation algorithms are described.

1.1.1 Least-Mean-Squares Algorithms

The objective function $E[e_k^2]$ of the optimal Wiener filter can be approximated by using a time average of e_k^2, i.e.,

$$\sigma_k^2 = \lambda\sigma_{k-1}^2 + (1-\lambda)e_k^2 \tag{1.1}$$

with $0 \leq \lambda < 1$ [1]. The basic least-mean-squares (LMS) algorithm [1, 2, 3, 4] uses (1.1) to approximate $E[e_k^2]$; however, it uses $\lambda = 0$ and, consequently, a simple adaptation algorithm is achieved [1]. The adaptation formula of the LMS algorithm is obtained by adding the negative of the gradient of the objective function in (1.1) to the weight vector $\boldsymbol{w}_{k-1} \in \mathcal{R}^{M\times 1}$, i.e.,

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} - \frac{\mu}{2}\frac{\partial e_k^2}{\partial \boldsymbol{w}_{k-1}} \tag{1.2}$$

where μ is the step size that controls the stability, convergence speed, and steady-state misalignment of the algorithm [1]. A larger value of μ yields faster convergence and increased steady-state misalignment and, on the other hand, a smaller value yields a slower convergence and reduced steady-state misalignment. To exploit the advantages of both a smaller and larger step size, several variable step-size LMS algorithms have been developed that keep the step size larger during transience and smaller during steady state [5, 6, 7]. The step size μ in (1.2) for the algorithms in [5], [6], and [7] is evaluated as

$$\mu_k = \alpha\mu_{k-1} + \gamma e_k^2, \tag{1.3}$$

$$\mu_k = \alpha\mu_{k-1} + \gamma p_k^2 \tag{1.4}$$

with

$$p_k = \beta p_{k-1} + (1-\beta)e_k e_{k-1}, \tag{1.5}$$

and

$$\mu_k = \mu_{max}\left(1 - exp^{-\alpha z_k}\right) \tag{1.6}$$

with

$$\begin{aligned} z_k &= f_k - 3p_k^2 && (1.7a)\\ f_k &= \beta f_{k-1} + (1-\beta)e_k^4 && (1.7b)\\ p_k &= \beta p_{k-1} + (1-\beta)e_k^2, && (1.7c) \end{aligned}$$

respectively, where α, β, and γ are tuning parameters chosen in the range 0 to 1. The variable step-size LMS algorithm in [6] is found to offer improved performance for low signal-to-noise ratios (SNRs). Low computational complexity and numerical robustness are the main advantages of LMS algorithms that make them most popular especially for applications in wireless communications [1, 2, 3].

Another class of algorithms known as *data-normalized* algorithms offer improved performance compared to the LMS algorithms at the cost of increased computational cost. This class of algorithms uses the step size in (1.2) along with data normalization as will be discussed in the next section.

1.1.2 Normalized Least-Mean-Squares Algorithms

The weight-vector update formula for the NLMS algorithm is obtained as

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} - \frac{\mu}{2\|\boldsymbol{x}_k\|^2}\frac{\partial e_k^2}{\partial \boldsymbol{w}_{k-1}} \tag{1.8}$$

with $0 < \mu < 2$ [1, 2, 3, 8, 9]. As can be seen, if the step size in (1.2) is made proportional to the inverse of the power of the input signal $\boldsymbol{x}_k$ we obtain the step size for the NLMS algorithm. In other words, like the LMS algorithms in [5, 6, 7] the NLMS algorithm also minimizes the objective function in (1.1) and hence it is a variable step size LMS algorithm. However, the computational complexity associated with the NLMS algorithm is increased relative to that in the LMS algorithms in [5, 6, 7]. Nonetheless, the NLMS algorithm based on (1.8) is commonly referred to in the literature as the NLMS algorithm with constant step size μ. As in the case of the LMS algorithm, improved performance can be achieved in the NLMS algorithm if the step size μ in (1.8) is kept large during transience and small during steady state.

A nonparametric NLMS (NPNLMS) algorithm was reported in [10] that uses (1.8) with a variable step size μ computed as

$$\mu_k = \begin{cases} 1 - \dfrac{\gamma}{\sigma_k} & \text{if } |e_k| > \gamma \\ 0 & \text{otherwise} \end{cases} \tag{1.9}$$

with $\gamma = \sqrt{\sigma_v^2}$ where σ_v^2 is the variance of the measurement noise, and parameter σ_k is obtained from (1.1) with $0 < \lambda < 1$. The NPNLMS algorithm in [10] offers significant improvement as compared to the conventional NLMS algorithm.

The main drawback of the LMS and the NLMS algorithms is their poor convergence speed for correlated input signals. In such situations, the affine projection algorithm discussed below can be a viable alternative.

1.1.3 Affine Projection Algorithms

Affine-projection (AP) algorithms offer superior convergence performance relative to LMS algorithms, especially for correlated input signals. However, they require a significantly increased amount of computational effort [2, 3]. The weight-vector update formula for the AP algorithm is

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} - \frac{\mu}{2}\frac{\partial J(\boldsymbol{e}_k)}{\partial \boldsymbol{w}_{k-1}}\boldsymbol{u}_1 \tag{1.10}$$

where $\boldsymbol{u}_1^T = [1\ 0\ \cdots\ 0]$ is a vector of dimension L, $J(\boldsymbol{e}_k) = \boldsymbol{e}_k^T(\delta\boldsymbol{I} + \boldsymbol{X}_k^T\boldsymbol{X}_k)^{-1}\boldsymbol{e}_k$ and $\boldsymbol{e}_k = \boldsymbol{d}_k - \boldsymbol{X}_k^T\boldsymbol{w}_{k-1} \in \mathcal{R}^{L\times 1}$ is the error signal vector [11]. Parameter L is known as the *projection order* of the algorithm [3]. Matrix $\boldsymbol{X}_k \in \mathcal{R}^{M\times L}$ is the input signal matrix obtained as $\boldsymbol{X}_k = [\boldsymbol{x}_k\ \boldsymbol{x}_{k-1}\ \cdots\ \boldsymbol{x}_{k-L+1}]$.

Several variants of the basic AP algorithm have been developed and analyzed in the literature [12, 13, 14, 15, 16, 17, 18]. The most recent developments on this class of algorithms can be found in [19, 20, 21, 22, 23, 24]. Improved performance can be achieved in AP algorithms by using a variable step size μ, a variable regularization parameter δ in $J(\boldsymbol{e}_k)$, and a variable projection order L. The algorithms in [24, 25, 26] use a variable L, the algorithm in [22] uses a variable regularization parameter δ, and the algorithms in [19, 20, 23] use a variable step size μ. The algorithm in [23] is designed exclusively for acoustic-echo-cancelation applications.

The step size μ in (1.10) for the variable step size AP (VSSAP) algorithm reported in [20] is given by

$$\mu_k = \mu_{max} \frac{\|\hat{\boldsymbol{q}}_k\|^2}{\|\hat{\boldsymbol{q}}_k\|^2 + C} \tag{1.11}$$

where

$$\hat{\boldsymbol{q}}_k = \alpha \hat{\boldsymbol{q}}_{k-1} + (1-\alpha)\boldsymbol{X}_k(\boldsymbol{X}_k^T \boldsymbol{X}_k)^{-1}\boldsymbol{e}_k \tag{1.12}$$

and $C = L/(10^{SNR/10})$. The VSSAP algorithm also yields a reduced steady-state misalignment for the same projection order compared to the AP algorithm [20].

An NLMS algorithm with variable step size is also possible [20] which is essentially the same as the AP algorithm with $L = 1$.

In the next section, we discuss the constrained AP algorithms reported in [27].

1.1.4 Constrained Affine Projection Algorithms

Linearly constrained adaptive filters have several applications in signal processing such as system identification, interference cancelation in direct-sequence code-division multiple access (DS-CDMA) communication systems, and array antenna beamforming [28, 29, 30, 31, 32, 33, 27]. The most widely used adaptation algorithms in such applications are the constrained least-mean-squares (CLMS) and the generalized sidelobe canceler least-mean squares (GSC-LMS) algorithms reported in [28] and [29], respectively, due to the simplicity of LMS algorithms. Constrained normalized LMS (CNLMS) and constrained binormalized data-reusing LMS (CBIDR-LMS) algorithms that perform better than LMS algorithms, especially when the input signal is correlated, have been proposed in [32]. Variable step-size CNLMS and CBIDR-LMS algorithms were proposed in [33] and [27], respectively. Later on constrained AP (CAP) algorithms with a constant and variable step size were reported in [27] for colored input signals.

The update formula for the CAP algorithm with a constant step size is obtained by using (1.10) as

$$\boldsymbol{w}_k = \boldsymbol{Z}\left[\boldsymbol{w}_{k-1} - \frac{\mu}{2}\frac{\partial J(\boldsymbol{e}_k)}{\partial \boldsymbol{w}_{k-1}}\boldsymbol{u}_{\boldsymbol{1}}\right] + \boldsymbol{F} \tag{1.13}$$

where $J(\boldsymbol{e}_k) = \boldsymbol{e}_k^T(\delta\boldsymbol{I} + \boldsymbol{X}_k^T\boldsymbol{Z}\boldsymbol{X}_k)^{-1}\boldsymbol{e}_k$,

$$\boldsymbol{Z} = \boldsymbol{I} - \boldsymbol{C}^T\left(\boldsymbol{C}\boldsymbol{C}^T\right)^{-1}\boldsymbol{C} \tag{1.14}$$

$$\boldsymbol{F} = \boldsymbol{C}^T\left(\boldsymbol{C}\boldsymbol{C}^T\right)^{-1}\boldsymbol{f} \tag{1.15}$$

with $\boldsymbol{C} \in \mathcal{R}^{N\times M}$ with $N < M$ and $\boldsymbol{f} \in \mathcal{R}^{N\times 1}$ are the constraint matrix and vector, respectively; all other parameters are the same as in (1.10).

In the next section, we discuss the set-membership (SM) filtering method based on which several variable step-size adaptation algorithms have been developed [3].

1.1.5 Set-Membership Algorithms

Conventional SM adaptive-filtering schemes estimate the weight vector $\boldsymbol{w}$ that would cause the magnitude of the output error

$$e = d - \boldsymbol{w}^T\boldsymbol{x} \tag{1.16}$$

to be less than or equal to a prespecified bound $\gamma \in \mathcal{R}_+$ for all possible input-desired signal pairs $(\boldsymbol{x},\ d)$ [3, 34]. The set of all possible input-desired signal pairs $(\boldsymbol{x},\ d)$ is commonly referred to as the *data space* and is denoted as S. The output error based on the SM adaptive-filtering (SMF) criterion in (1.16) must satisfy the condition

$$|e|^2 \leq \gamma^2 \quad \forall\ (\boldsymbol{x},\ d) \in S \tag{1.17}$$

The set of all possible vectors $\boldsymbol{w}$ that satisfy (1.17) whenever $(\boldsymbol{x},\ d) \in S$, designated as Θ, is referred to as the *feasibility* or *solution set* and can be expressed as

$$\Theta = \cap_{(\boldsymbol{x},\ d)\in S}\{\boldsymbol{w} \in \mathcal{R}^M :\ |d - \boldsymbol{w}^T\boldsymbol{x}| \leq \gamma\} \tag{1.18}$$

If the adaptive filter is trained with k input-desired data pairs $\{\boldsymbol{x}_i,\ d_i\}_{i=1}^k$, then the set containing all vectors $\boldsymbol{w}$ for which the associated output error at iteration k is consistent with (1.17) is called the *constraint* or *observation set.* It is given by

$$H_k = \{\boldsymbol{w} \in \mathcal{R}^M :\ |d_k - \boldsymbol{w}^T\boldsymbol{x}_k| \leq \gamma\} \tag{1.19}$$

The intersection of the constraint sets H_k over all iterations $i = 1, 2, \ldots, k$ is called the *exact membership set* and is given by

$$\Psi_k = \cap_{i=1}^{k} H_i \tag{1.20}$$

Evidently, the feasibility set Θ is a subset of the exact membership set Ψ_k in any given iteration.

Based on this approach several set-membership adaptation algorithms have been developed in [19, 34, 35]. The set-membership NLMS (SMNLMS) reported in [34] uses the update formula in (1.8) where the step size is computed as

$$\mu_k = \begin{cases} 1 - \dfrac{\gamma}{|e_k|} & \text{if } |e_k| > \gamma \\ 0 & \text{otherwise} \end{cases} \tag{1.21}$$

with $\gamma = \sqrt{5\sigma_v^2}$. Another set-membership LMS type adaptation algorithm was reported in [36] where the weight vector in (1.8) is projected onto an adaptive convex set $\boldsymbol{A}_k$ in such a way that the projected weight vector is closer to that of the unknown system. The convex set $\boldsymbol{A}_k$ at iteration k is obtained by using the optimal bounding ellipsoid algorithm in [37] and it yields robust performance even when the input signal lacks persistent excitation and the perturbations in the unknown system are small. However, this algorithm cannot operate in real time, i.e., a subloop is required to get the convex set $\boldsymbol{A}_k$. Similarly in [38] a gradient projection optimal bounding ellipsoid algorithm is reported for channel equalization applications where the equalizer parameters are computed by projecting the gradient estimate onto a set that is updated using *a priori* information on the instantaneous error magnitude and is shown to offer robust performance when the input signal lacks persistent excitation. The SMNLMS algorithm yields a significantly reduced steady-state misalignment compared to the NLMS algorithm for the same convergence speed and also it requires reduced computational load. Similarly, the set-membership AP (SMAP) algorithm reported in [19] uses the update formula in (1.10) with the size μ given by (1.21). This algorithm was referred to as the SMAP algorithm in [19] but was later referred to as the *simplified* SMAP algorithm in [39]. For the sake of consistency with the more recent publication on this algorithm, namely, [39], we will refer to this algorithm as the simplified SMAP (SSMAP) algorithm hereafter. The SSMAP algorithm yields reduced steady-state misalignment relative to the AP algorithm for the same projection order. The algorithms reported in [25], [26] are variants of the SSMAP algorithm in [19]; the algorithm in [25] yields improved convergence speed for sparse system-identification applications and the algorithm in [26] yields a slightly improved steady-state misalignment as compared to the SSMAP algorithm in [19, 39]. The set-membership binormalized data-reusing LMS algorithm in [35] is an alternative implementation of the SSMAP algorithm in [19] with a projection order of two. Some variants of the SM algorithms that estimate the error bound γ during the learning stage are reported in [40, 41, 42].

In the next section, we discuss the class of linearly constrained SM algorithms.

1.1.6 Linearly Constrained Set-Membership Algorithms

Like the SM criterion, the linearly constrained SM criterion satisfies the condition in (1.17) in addition to the constraint

$$\boldsymbol{C}\boldsymbol{w} = \boldsymbol{f} \quad \forall \, (\boldsymbol{x}, \ d) \in S \tag{1.22}$$

where $\boldsymbol{C} \in \mathcal{R}^{N \times M}$ with $N < M$ and $\boldsymbol{f} \in \mathcal{R}^{N \times 1}$ are the constrained matrix and vector, respectively, and S denotes the data space [27]. The solution set is expressed as

$$\Theta = \cap_{(\boldsymbol{x}, \ d) \in S} \{\boldsymbol{w} \in \mathcal{R}^M : \ |d - \boldsymbol{w}^T\boldsymbol{x}| \leq \gamma \text{ and } \boldsymbol{C}\boldsymbol{w} = \boldsymbol{f}\} \tag{1.23}$$

which contains all the $\boldsymbol{w}$ that satisfy (1.17) and (1.22) [27]. The constraint set at iteration k is given by

$$H_k = \{\boldsymbol{w} \in \mathcal{R}^M : \ |d_k - \boldsymbol{w}^T\boldsymbol{x}_k| \leq \gamma \ \text{ and } \boldsymbol{C}\boldsymbol{w} = \boldsymbol{f}\} \tag{1.24}$$

The intersection of the constraint sets H_k over all iterations $i = 1, \ 2, \ldots, k$ is given by

$$\Psi_k = \cap_{i=1}^{k} H_i \tag{1.25}$$

Evidently, the feasibility set Θ is a subset of the exact membership set Ψ_k in any given iteration.

Using μ_k given by (1.21) in (1.13), the update formula of the set-membership CAP (SMCAP) algorithm in [27] is obtained. The CAP and SMCAP algorithms reported in [27] achieve a higher convergence speed and similar misalignment as compared to those of the CLMS and CNLMS algorithms.

The algorithms we discussed so far belong to the steepest-descent family. When the input signal is highly colored or bandlimited, LMS algorithms as well as other algorithms of the steepest-descent family converge slowly and the capability of such algorithms in tracking nonstationarities deteriorates. In such situations, more sophisticated algorithms that belong to the Newton family are preferred.

In the next two sections, the two most important Newton-type adaptation algorithms, namely, the recursive least-squares (RLS) and quasi-Newton (QN) algorithms are discussed.

1.1.7 Recursive Least-Squares Algorithms

RLS algorithms are different from the algorithms based on the MSE criterion in the sense that they do not use (1.1) to estimate $E[e_k^2]$. These algorithms optimize the objection function [1, 2, 3]

$$J = \sum_{i=1}^{k} \lambda^{k-i}(d_i - \boldsymbol{w}_k^T \boldsymbol{x}_i)^2 \tag{1.26}$$

to obtain the weight vector at iteration k as

$$\boldsymbol{w}_k = \boldsymbol{R}_k^{-1} \boldsymbol{p}_k \tag{1.27}$$

where $\boldsymbol{R}_k$ and $\boldsymbol{p}_k$ denote the autocorrelation matrix and crosscorrelation vector, respectively, defined as

$$\boldsymbol{R}_k = \lambda \boldsymbol{R}_{k-1} + \boldsymbol{x}_k \boldsymbol{x}_k^T \tag{1.28}$$

and

$$\boldsymbol{p}_k = \lambda \boldsymbol{p}_{k-1} + \boldsymbol{x}_k d_k \tag{1.29}$$

RLS algorithms offer the fastest convergence and the lowest steady-state misalignment relative to other types of algorithms. However, the computational complexity of these algorithms is of order M^2, denoted as $\mathcal{O}(M^2)$, which is much more than that of LMS, NLMS, and AP algorithms which have computational complexity of $\mathcal{O}(M)$. Some fast RLS (FRLS) algorithms with computational complexities of $\mathcal{O}(M)$ can be found in [43, 44, 45, 46, 47]. RLS algorithms based on the QR decomposition (QRD) of $\boldsymbol{R}_k$ can be found in [47]. The computational complexity of these algorithms is of $\mathcal{O}(M^2)$. Some fast QRD (FQRD) algorithms with computational complexity of $\mathcal{O}(M)$

can be found in [48, 49, 50, 51, 52, 53, 54]. FRLS and FQRD algorithms suffer from increased numerical instability problems due to their inheritance of the numerical instability problems of their parent RLS and QRD algorithms, respectively, and the simplifications used to obtain these algorithms [49]. However, the FQRD algorithm reported in [49] offers numerically stable operation in low-precision implementations and in the absence of persistent excitation. The numerical instability problem associated with the conventional RLS algorithm is discussed in [55, 56] and in [56] an upper bound on the relative precision to assure the bounded-input bounded-output (BIBO) stability of the algorithm in stationary and nonstationary environments is derived. Formulas for choosing the forgetting factor to avoid explosive divergence for a given precision in the conventional RLS algorithm are given in [56]. However, these formulas were derived on the assumption that the input signal is persistently exciting. Furthermore, the input-signal statistics must be known *a priori* in order to use these formulas. Consequently, a prudent strategy for the derivation of fast Newton-type algorithms would be to start with a parent algorithm that is inherently stable. A viable alternative to achieve numerically robust fast QN algorithms would be to use the quasi-Newton (QN) algorithms reported in [57, 58], which offer better numerical robustness than RLS algorithms. The LMS-Newton (LMSN) algorithms reported in [59] offer better convergence performance than the conventional RLS algorithm. Improved versions of the LMSN algorithms are reported in [60]. In addition to the numerical instability problems, RLS algorithms suffer from a reduced re-adaptation capability and high sensitivity to impulsive noise.

Known approaches for improving the performance of adaptive filters in impulsive-noise environments involve the use of nonlinear clipping [61, 62], robust statistics [63, 64, 65], or order statistics. The common step in the adaptive filters reported in [61, 62, 63, 64, 65] is to detect the presence of impulsive noise by comparing the magnitude of the error signal with the value of a threshold parameter which is a scalar multiple of the variance of the error signal and then either stop or reduce the learning rate of the adaptive filter. The adaptation algorithms in [61, 65] use the Huber mixed-norm M-estimate objective function [66] and the algorithms in [63, 64] use the Hampel three-part redescending M-estimate objective function [66]. The nonlinear recursive least-squares (NRLS) algorithm in [62] uses nonlinear clipping to control the learning rate and offers better performance in impulsive-noise environments than the conventional RLS algorithm. The recursive least-mean (RLM) algorithm reported in [63] offers faster convergence and better robustness than the NRLS algorithm. The

RLM algorithm uses the autocorrelation matrix

$$\boldsymbol{R}_k = \lambda \boldsymbol{R}_{k-1} + q(e_k)\boldsymbol{x}_k\boldsymbol{x}_k^T \tag{1.30}$$

and the crosscorrelation vector

$$\boldsymbol{p}_k = \lambda \boldsymbol{p}_{k-1} + q(e_k)\boldsymbol{x}_k d_k \tag{1.31}$$

to obtain the weight vector $\boldsymbol{w}_k$ in (1.27) where $q(e_k)$ is given by

$$q(e_k) = \begin{cases} 1 & \text{for } 0 < |e_k| \leq \xi \\ \dfrac{\xi}{|e_k|} & \text{for } \xi < |e_k| \leq \Delta_1 \\ \left(1 - \dfrac{\Delta_2}{|e_k|}\right)\dfrac{\xi}{\Delta_1 - \Delta_2} & \text{for } \Delta_1 < |e_k| \leq \Delta_2 \\ 0 & \text{otherwise} \end{cases} \tag{1.32}$$

Parameters ξ, Δ_1, and Δ_2 are chosen as $1.96\hat{\sigma}_{e,k}$, $2.24\hat{\sigma}_{e,k}$, and $2.576\hat{\sigma}_{e,k}$, respectively, where $\hat{\sigma}_{e,k}^2$ is an estimate of the variance of the *a priori* error signal e_k. Variance $\hat{\sigma}_{e,k}^2$ is estimated as

$$\hat{\sigma}_{e,k}^2 = \lambda\hat{\sigma}_{e,k-1}^2 + c_1(1-\lambda)\text{med}(\boldsymbol{a}_k) \tag{1.33}$$

where *med* denotes the median operation, $\boldsymbol{a}_k = \begin{bmatrix} e_k^2 \ e_{k-1}^2 \ \cdots \ e_{k-P+1}^2 \end{bmatrix}$ is a vector of dimension P, $0 < \lambda < 1$ is the *forgetting factor*, and $c_1 = 1.483[1 + 5/(P-1)]$ is the *finite-sample correction factor* alluded to in [66]. A fast RLM algorithm that has a computational complexity of $\mathcal{O}(M)$ is reported in [67]. For an impulsive-noise corrupted e_k, $q(e_k)$ is expected to assume a zero value that would prevent adaptations in (1.30) and (1.31) and hence an impulsive noise-corrupted e_k would have no consequence on $\boldsymbol{w}_k$ in (1.27). However, an impulsive noise-corrupted $\boldsymbol{x}_k$ does not necessarily cause an impulsive noise corrupted $e_k = d_k - \boldsymbol{w}_{k-1}^T\boldsymbol{x}_k$ since the impulsive noise component in $\boldsymbol{x}_k$ can be filtered out if $\boldsymbol{w}_{k-1}$ has an amplitude spectrum that resembles the amplitude response of a lowpass filter.

Another RLS algorithm that bounds the L_2 norm of the difference of the weight vector $\|\boldsymbol{w}_k - \boldsymbol{w}_{k-1}\|^2$ by an upper bound δ to suppress the effect of the impulsive noise corrupted desired or input signal in the weight vector update formula was reported in [68]. This algorithm has a computational complexity of $\mathcal{O}(M)$. Actually in [68], the instantaneous power of the scaled error signal is lowpass filtered and then used to switch the step size of the algorithm between two levels one of which suppresses the error signal corrupted by impulsive noise during the adaptation of the weight vector.

The robust algorithms in [63, 67, 68] belong to the RLS family and hence they converge significantly faster than algorithms of the steepest-descent family [3].

1.1.8 Quasi-Newton Algorithms

The quasi-Newton (QN) algorithms in [57, 58, 69] use the weight-vector update formula

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} - \frac{\mu}{2}\boldsymbol{S}_{k-1}\frac{\partial e_k^2}{\partial \boldsymbol{w}_{k-1}} \tag{1.34}$$

where $\boldsymbol{S}_{k-1} \in \mathcal{R}^{M\times M}$ is a positive definite matrix. As can be seen, using $\boldsymbol{S}_{k-1} = \boldsymbol{I}$ in (1.34) results in (1.2). In other words, like LMS algorithms, QN algorithms use the objective function e_k^2 to obtain the weight vector update formula in (1.34). However, the search direction in the QN algorithms is modified by using a positive-definite matrix that is not equal to the identity matrix, i.e., $\boldsymbol{S}_{k-1} \neq \boldsymbol{I}$. The gradient of e_k^2, i.e., $\partial e_k^2/\partial \boldsymbol{w}_{k-1}$, is used in the rank-one update formula [70] to obtain $\boldsymbol{S}_{k-1}$ for the QN algorithms in [57, 58, 69]. QN algorithms were found to be numerically more robust in fixed- and floating-point arithmetic implementations for ill-conditioned input signals compared to the RLS algorithm described in [57, 58, 69]. However, QN algorithms are not robust with respect to impulsive noise. The QN algorithm in [64] uses the weight-vector update formula

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} - \frac{\mu}{2}\boldsymbol{S}_{k-1}\frac{\partial J(e_k)}{\partial \boldsymbol{w}_{k-1}} \tag{1.35}$$

where the objective function $J(e_k)$ is given by

$$J(e_k) = \begin{cases} e_k^2 & \text{for } 0 < |e_k| \leq \xi \\ 2\xi|e_k| - \xi^2 & \text{for } \xi < |e_k| \leq \Delta_1 \\ \xi(\Delta_2 + \Delta_1) - \xi^2 + \xi\dfrac{(|e_k| - \Delta_2)^2}{\Delta_1 - \Delta_2} & \text{for } \Delta_1 < |e_k| \leq \Delta_2 \\ \xi(\Delta_2 + \Delta_1) - \xi^2 & \text{otherwise} \end{cases} \tag{1.36}$$

The gradient of $J(e_k)$, i.e., $\partial J(e_k)/\partial \boldsymbol{w}_{k-1}$, is used in the self-scaling variable matrix update formula in [71] to obtain $\boldsymbol{S}_{k-1}$ in (1.35). Parameters ξ, Δ_1, and Δ_2 are set to the same values as in the RLM algorithm. For an impulsive-noise corrupted e_k, the objective function $J(e_k)$ assumes the value for the range $|e_k| > \Delta_2$ in (1.36) where the gradient $\partial J(e_k)/\partial \boldsymbol{w}_{k-1}$ assumes a zero value and, therefore, no update is performed in (1.35). In this way, robust performance with respect to impulsive noise is achieved in the QN algorithm described in [64].

A constrained version of the QN algorithm described in [58], referred to as the constrained QN (CQN) algorithm, can be found in [72]. A fast QN (FQN) algorithm that has a computational complexity of $\mathcal{O}(M)$ was reported in [73].

1.1.9 Minimum Error-Entropy Algorithms

The minimum error-entropy criterion was proposed in [74, 75] as an alternative to the MSE criterion. Since the MSE criterion takes into account only the second-order statistics of a signal it yields optimal performance for applications where signals can be modeled in terms of Gaussian distributions. In other applications, improved performance can be achieved by using the minimum error-entropy criterion as it uses higher-order statistics [74, 75]. Based on this criterion, the stochastic minimum error-entropy (MEE), minimum-error entropy with self-adjusting step size (VMEE), and the normalized minimum-error entropy (NMEE) algorithms were proposed in [76], [77], and [78], respectively. The operation of the MEE algorithms is based on the minimization problem

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad E_2(e) = -\log \textstyle\int_{-\infty}^{\infty} f_e^2(e)\, de \tag{1.37}$$

where $E_2(e)$ is Renyi's entropy [79] of random error signal e with probability density function $f_e(e)$ [79]. Using the Parzen window with a Gaussian kernel $\kappa_\sigma(e)$ for estimating $f_e(e)$, the optimization problem in (1.37) can be expressed as

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad E_2(e) = -\log V(e) \tag{1.38}$$

where

$$V(e) = \frac{1}{N^2} \sum_{j=1}^{N} \sum_{i=1}^{N} \kappa_{\sigma\sqrt{2}}(e_j - e_i) \tag{1.39}$$

$V(e)$ is known as the *information potential* (IP) function of the error signal. The minimization problem in (1.38) is equivalent to maximizing the IP function in (1.39) since $\log V(e)$ is a monotonically increasing function. For a real-time implementation, a stochastic IP function

$$V_k(e) \approx \frac{1}{L} \sum_{i=k-L}^{k-1} \kappa_{\sigma\sqrt{2}}(e_k - e_i) \tag{1.40}$$

is maximized by using the update equation

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \mu \frac{\partial V_k(e)}{\partial \boldsymbol{w}} \tag{1.41}$$

where μ is the step size. An algorithm based on this updating formula is referred to as the MEE algorithm in [76].

If the IP function in (1.39) or (1.40) is a δ-distributed random variable, i.e., it achieves its maximum value if $e_k = 0$ for all k, then $V(e) \leq V(0)$ where $V(0) = 1/\sigma\sqrt{2\pi}$ is the upper bound on the achievable $V(e)$. Therefore, the maximization of $V_k(e)$ in (1.40) is equivalent to solving the optimization problem

$$\underset{\boldsymbol{w}}{\text{minimize}} \quad \|V(0) - V_k(e)\|^2 \tag{1.42}$$

The weight update equation that solves the problem in (1.42) is referred to as the VMEE recursion formula in [77] and it is given by

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \eta^* \frac{\partial V_k(e)}{\partial \boldsymbol{w}} \tag{1.43}$$

where $\eta^* = \mu[V(0) - V_k(e)]$ is the variable step size. The NMEE algorithm in [78], on the other hand, solves the optimization problem

$$\underset{\boldsymbol{w}}{\text{minimize}} ||\boldsymbol{w}_k - \boldsymbol{w}_{k-1}||^2 \tag{1.44}$$

subject to the constraint

$$V(e_{p,k}) - V(0) = 0$$

by using the weight update formula

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \mu \frac{e_k \nabla V(e_{p,k})}{\nabla V(e_{p,k})^T \boldsymbol{x}_k} \tag{1.45}$$

where $e_k = d_k - \boldsymbol{w}_{k-1}^T \boldsymbol{x}_k$ is the *a priori* error

$$\nabla V(e_{p,k}) = \frac{1}{2L\sigma^2} \sum_{i=k-L}^{k-1} (e_{p,k} - e_{p,i})\, \kappa_\sigma(e_{p,k} - e_{p,i})\, (\boldsymbol{x}_k - \boldsymbol{x}_i) \tag{1.46}$$

and $e_{p,i} = d_i - \boldsymbol{w}_k^T \boldsymbol{x}_i$ is the *a posteriori* error for $k-L \le i \le k$. The update equation in (1.45) requires $e_{p,i}$ in every iteration and to avoid this added complexity, the authors in [78] proposed replacing $e_{p,i}$ by $e_i = d_i - \boldsymbol{w}_{k-1}^T \boldsymbol{x}_i$ for $k - L \le i \le k$.

The VMEE and the conventional NMEE algorithms yield faster convergence than the MEE algorithm for the same misadjustment level [78]. The kernel size and step size need to be adjusted as the input power changes in both the MEE and the VMEE algorithms [78]. The conventional NMEE algorithm is less sensitive to the kernel size and input signal power and, therefore, yields improved performance compared to the MEE and the VMEE algorithms [78].

In the next section, we describe the so-called *iterative shrinkage* method which has been used for signal denoisng applications [80, 81]. This method can also be used to develop adaptation algorithms as will be demonstrated in chapter 8.

1.1.10 Iterative Shrinkage Method

Suppose the observed signal $\boldsymbol{o}_k$ is obtained as

$$\boldsymbol{o}_k = \boldsymbol{s}_k + \boldsymbol{v}_k \tag{1.47}$$

where $\boldsymbol{s}_k$ is the signal of interest and $\boldsymbol{v}_k$ is a contaminating white Gaussian noise signal. If we wish to recover $\boldsymbol{s}_k$ from $\boldsymbol{o}_k$, we have to filter out the noise signal $\boldsymbol{v}_k$. Since $\boldsymbol{v}_k$ is a white Gaussian noise signal, we cannot recover $\boldsymbol{s}_k$ from $\boldsymbol{o}_k$ by using a digital filter as the spectrum of $\boldsymbol{v}_k$ is spread over the entire frequency spectrum of $\boldsymbol{s}_k$. In the iterative shrinkage method, the solution of the minimization problem

$$\underset{\boldsymbol{b}_k}{\text{minimize}} \quad t\|\boldsymbol{b}_k\|_1 + \|\boldsymbol{D}\boldsymbol{b}_k - \boldsymbol{o}_k\|^2 \tag{1.48}$$

where $\boldsymbol{D}$ is a matrix whose columns form an orthonormal basis and t is a threshold parameter is used to obtain the signal of interest as $\hat{\boldsymbol{o}}_k = \boldsymbol{D}^T \boldsymbol{b}_{opt}$. For an appropriate value of t, $\hat{\boldsymbol{o}}_k$ can be very close to $\boldsymbol{s}_k$. The choice of t depends on the statistical characteristics of signal $\boldsymbol{s}_k$.

1.2 Original Contributions

The goal of this work has been to develop adaptation algorithms that (1) are robust with respect to impulsive noise, (2) yield reduced steady-state misalignment, (3) have high convergence speed, (4) have good tracking, and (5) entail reduced computational load. Several improved adaptation algorithms have been proposed as detailed below.

In chapter 2, we propose a robust set-membership affine projection (RSMAP) adaptation algorithm [82] that uses two error bounds. The RSMAP algorithm has two variants: one with a fixed threshold and the other with a variable threshold. The two algorithms offer similar performance and both of them offer more robust performance with respect to impulsive noise without sacrificing the convergence speed and tracking capability compared to the affine projection (AP) and simplified set-membership affine projection (SSMAP) algorithms reported in [11] and [19], respectively. In addition, they offer significantly reduced steady-state misalignment compared to the AP and SSMAP algorithms. The performance of SSMAP algorithms depends on the proper choice of the error bound [19] which in some applications cannot be prespecified accurately. The RSMAP algorithm with variable threshold eliminates this problem as it estimates the error bound during the learning process. These features of the proposed RSMAP algorithm are demonstrated in network-echo-path and acoustic-echo-path-identification applications for various levels of signal-to-noise ratio (SNR) and projection order. In addition, a practical acoustic-echo-cancelation application is considered to demonstrate the superior performance of the proposed RSMAP algorithms. An approximate steady-state MSE analysis is also described which is verified using simulation results obtained in a system-identification application.

In chapter 3, we propose a constrained robust set-membership affine projection (PCSMAP) algorithm [83]. Like the proposed RSMAP algorithm, the CRSMAP algorithm also works with two error bounds by which it yields a significantly reduced steady-state misalignment without compromising the convergence speed and tracking capability and at the same time it yields robust performance with respect to impulsive noise. These features of the RCSMAP algorithm are demonstrated by using simulation results in a linear-phase plant-identification application, a constrained time-series filtering application, and interference suppression in a DS-CDMA mobile communication system.

In chapter 4, we propose a novel robust recursive least-squares (PRRLS) adaptation algorithm that entails bounding the L_1 norm of the cross-correlation vector by a time-varying upper bound which in effect bounds the L_1 norm of the input-signal autocorrelation matrix and the L_2 norm of the weight vector [84]. The PRRLS algorithm has two variants: one for stationary environments and the other for non-stationary environments. Both of these algorithms bound the L_1 norm of the cross-correlation vector for an impulsive-noise-corrupted desired signal and thereby achieve robust performance with respect to impulsive noise. The algorithm for nonstationary environments also uses a time-varying forgetting factor that effectively tracks the nonstationarities of the unknown system. These features of the proposed algorithm are demonstrated by using simulation results in a system-identification application.

In chapter 5, we apply certain modifications to the known QN algorithm (KQN) reported in [57, 58] which lead to improved convergence speed and steady-state misalignment without sacrificing the most essential features of the KQN algorithm, i.e., numerical robustness as observed in [58]. Like the KQN algorithm, the proposed QN (PQN) algorithm [85] also uses the classical rank-one update formula [71] to obtain the inverse of the Hessian matrix. However, unlike the KQN algorithm, the PQN algorithm does not omit certain steps of the basic classical QN optimization algorithm [70]. The PQN algorithm performs data selective adaptation in the weight-vector update formula and the inverse of the Hessian matrix. It yields a reduced steady-state misalignment in both fixed- and floating-point implementations and converges much faster than the KQN algorithm for medium and high SNRs. A stability analysis shows that the PQN algorithm is asymptotically stable. A steady-state MSE analysis is also presented for the PQN algorithm which is then verified using simulation results obtained in a system-identification application.

In Chapter 6, we propose a robust QN (PRQN) adaptation algorithm [86] as an alternative to the known robust QN (KRQN) algorithm reported in [64]. The PRQN algorithm has two variants: one with fixed threshold and the other with variable threshold. The PRQN algorithm with variable threshold can be applied where the error bounds cannot be prespecified. The performance of the two algorithms is similar. The asymptotic stability of the PRQN algorithm is established. Expressions for the steady-state MSE for the cases of stationary and nonstationary environments are derived by using an energy-conservation relation reported in [87]. Like the RSMAP algorithm proposed in chapter 2, the PRQN algorithm also uses two error bounds, one of which increases if the amplitude of the error signal is increased. Therefore, we would get a large error bound for an impulsive-noise-corrupted error signal which would suppress the effect of impulsive noise in the learning process. The other error bound controls the initial convergence speed and tracking capability. In this way, a robust performance with respect to impulsive noise is achieved while retaining the fast initial convergence speed and good tracking capability. The PRQN algorithm also yields a reduced steady-state misalignment as compared to the KQN and KRQN algorithms. Simulation results in system-identification applications in stationary as well as nonstationary environments are used to demonstrate the superior convergence characteristics of the PRQN algorithm as compared to the KQN and the KRQN algorithms.

In chapter 7, we propose a new normalized minimum-error entropy (PNMEE) adaptation algorithm [88] as an alternative to the stochastic minimum-error entropy (MEE), normalized MEE (NMEE), and self-scaling variable step size MEE (VMEE) algorithms reported in [89], [78], and [77], respectively. The proposed NMEE algorithm offers fast convergence and reduced steady-state misalignment as compared to the MEE, NMEE, and VMEE algorithms by using extensive simulation results.

In chapter 8, we propose a family of so-called *shrinkage adaptation* algorithms. The step size used in these algorithms is obtained by minimizing the power of the noise-free *a posteriori* error signal by solving a one-dimensional minimization problem. Information about the noise-free *a priori* error signal is obtained by using the iterative shrinkage method described in [80, 81]. Based on this method, a shrinkage AP (SHAP) algorithm, a shrinkage NLMS (SHNLMS) algorithm, and a shrinkage LMS (SHLMS) algorithm are proposed [90]. The superior convergence characteristics of the proposed algorithms with respect to other competing algorithms are demonstrated by using simulation results in system identification and echo-cancelation applications.

Finally, in chapter 9, we draw conclusions and make recommendations for future research.

Chapter 2

Robust Set-Membership Affine Projection Adaptation Algorithm

2.1 Introduction

Performance analyses of the affine projection (AP) and simplified set-membership AP (SSMAP) algorithms are presented in [20] and [39], respectively. The analysis presented in [20] shows that the convergence speed of the AP algorithm increases as the projection order is increased, at the expense of increased steady-state misalignment. The same conclusion was also drawn for the SSMAP algorithm in [39]. However, by using a variable step size, the SSMAP algorithm yields reduced steady-state misalignment relative to that of the AP algorithm for the same projection order [19]. The prespecified error bound in the SSMAP algorithm is usually chosen as $\sqrt{5}\sigma_v$, where σ_v^2 is the variance of the measurement noise, in order to achieve a good balance between convergence speed and computational effort [19, 25, 26, 39]. In practice, however, it may not be possible to accurately specify the error bound in the SSMAP algorithm. In addition, as for the AP algorithm, the performance of the SSMAP algorithm is affected by outliers in the error signal samples that can be brought about by impulsive-noise interference. It is, therefore, of interest to develop a SMAP algorithm whose (1) performance remains largely insensitive to outliers brought about by impulsive noise, (2) sensitivity of the steady-state misalignment on the projection order is significantly reduced, (3) re-adaptation capability is preserved, and (4) sensitivity of the convergence performance on the proper choice of error bound is significantly reduced.

In this chapter, we develop a new SMAP adaptation algorithm that uses two error bounds [82]. Both of the error bounds are estimated by using the power of the error signal during the learning stage. One of the two error bounds yields faster convergence and good re-adaptation capability whereas the other yields a reduced steady-state misalignment and suppresses impulsive-noise interference.

2.2 Robust Set-Membership Affine Projection Algorithm

The proposed robust SMAP (RSMAP) algorithm performs weight adaptation at iteration k such that the updated weight vector belongs to the constraint sets at the L most recent iterations, i.e., $\boldsymbol{w}_k \in \Psi_k^L$ in (1.20). Whenever the weight vector $\boldsymbol{w}_{k-1}$ is not a member of Ψ_k^L, an update is performed by solving the optimization problem [82]

$$\begin{aligned} \underset{\boldsymbol{w}_k}{\text{minimize}} \quad & \|\boldsymbol{w}_k - \boldsymbol{w}_{k-1}\|^2 \\ \text{subject to :} \quad & \boldsymbol{d}_k - \boldsymbol{X}_k^T \boldsymbol{w}_k = \boldsymbol{g}_k \end{aligned} \tag{2.1}$$

where $\boldsymbol{d}_k \in \mathcal{R}^{L\times 1}$ is the desired signal vector, $\boldsymbol{g}_k \in \mathcal{R}^{L\times 1}$ is the error-bound vector, $\boldsymbol{X}_k \in \mathcal{R}^{M\times L}$ is the input signal matrix, i.e., $\boldsymbol{X}_k = [\boldsymbol{x}_k \ \boldsymbol{x}_{k-1} \ \cdots \ \boldsymbol{x}_{k-L+1}]$. The solution of the problem in (2.1) results in the update formula

$$\boldsymbol{w}_k = \begin{cases} \boldsymbol{w}_{k-1} + \boldsymbol{X}_k (\boldsymbol{X}_k^T \boldsymbol{X}_k)^{-1} (\boldsymbol{e}_k - \boldsymbol{g}_k) & \text{if } |e_k| > \gamma \\ \boldsymbol{w}_{k-1} & \text{otherwise} \end{cases} \tag{2.2}$$

where $e_k = d_k - \boldsymbol{x}_k^T \boldsymbol{w}_{k-1}$ is the *a priori* error, $\boldsymbol{e}_k = [e_k \ \epsilon_{k-1} \cdots \epsilon_{k-L+1}]^T$, and $\epsilon_{k-i} = d_{k-i} - \boldsymbol{x}_{k-i}^T \boldsymbol{w}_{k-1}$ is the *a posteriori* error at iteration $k-1$. Choosing the error bound vector $\boldsymbol{g}_k$ in (2.2) as

$$\boldsymbol{g}_k^T = \gamma \left[\frac{e_k}{|e_k|} \ \frac{\epsilon_{k-1}}{|e_k|} \ \cdots \ \frac{\epsilon_{k-L+1}}{|e_k|} \right] \tag{2.3}$$

leads to the update formula

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \alpha_k \boldsymbol{X}_k (\boldsymbol{X}_k^T \boldsymbol{X}_k)^{-1} \boldsymbol{e}_k \tag{2.4}$$

where

$$\alpha_k = \begin{cases} 1 - \dfrac{\gamma}{|e_k|} & \text{if } |e_k| > \gamma \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

For this error bound, if $|e_k| > g_k(1)$, we obtain $|\epsilon_{k-i}| > g_k(i)$ for $i = 2, \ldots, L$ where $g_k(i)$ denotes the ith element of vector $\boldsymbol{g}_k$.

In the SSMAP adaptation algorithm, the error-bound vector $\boldsymbol{g}_k$ is chosen as $\boldsymbol{g}_k = [\gamma \text{sign}(e_k)\ \epsilon_{k-1}\ \cdots\ \epsilon_{k-L+1}]^T$ by which the recursion formula in (2.2) simplifies to

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \alpha_k \boldsymbol{X}_k (\boldsymbol{X}_k^T \boldsymbol{X}_k)^{-1} e_k \boldsymbol{u}_1 \tag{2.6}$$

where α_k is defined in (2.5) and $\boldsymbol{u}_1 = [1\ 0\ \cdots\ 0]^T$ [19]. Using projection order $L = 2$ and the closed-form inverse of $\boldsymbol{X}_k^T \boldsymbol{X}_k$ in (2.6), the weight-vector update formula becomes identical to that of the SM binormalized data-reusing LMS-II adaptation algorithm in [35]. The AP algorithm (APA) used in [20], on the other hand, uses the update formula

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \mu \boldsymbol{X}_k (\boldsymbol{X}_k^T \boldsymbol{X}_k)^{-1} \boldsymbol{e}_k \tag{2.7}$$

where μ is a fixed step size and $0 < \mu < 1$. As can be seen, the proposed RSMAP algorithm based on (2.4) can be considered as a variable step-size AP algorithm in which the step size α_k varies in the range zero to unity.

Two versions of the proposed RSMAP algorithm are possible, one with fixed and the other with variable threshold, as detailed below. These will be referred to as the RSMAP1 and RSMAP2 algorithms, respectively.

2.2.1 RSMAP1 Algorithm

In the proposed RSMAP1 algorithm, factor γ in the error-bound vector in (2.3) is chosen as

$$\gamma_k = \begin{cases} \|\boldsymbol{e}_k\|_\infty - \nu\theta_k & \text{if } \|\boldsymbol{e}_k\|_\infty > \theta_k \\ \gamma_c & \text{otherwise} \end{cases} \tag{2.8}$$

where $0 < \nu \ll 1$, $\gamma_c = \sqrt{5\sigma_v^2}$ [19] and θ_k is chosen to be less than γ_c in order to assure that the algorithm will work with error bound $\|\boldsymbol{e}_k\|_\infty - \nu\theta_k$ during steady state and also for an impulsive-noise corrupted e_k. Extensive simulations have shown that a suitable value for θ_k can be obtained as $\theta_k = Q\sigma_{1,k}$ where $1.86 \leq Q \leq 1.98$. The value $Q = 1.88$ was found to lead to improved computational efficiency. The variance of the error signal, $\sigma_{1,k}^2$, is obtained as

$$\begin{aligned} \mathcal{C} &= \text{median}(\boldsymbol{c}_k) & (2.9) \\ \sigma_{1,k}^2 &= \lambda\sigma_{1,k-1}^2 + (1-\lambda)\mathcal{C} & (2.10) \end{aligned}$$

where $\boldsymbol{c}_k = [e_k^2 + \epsilon \ e_{k-1}^2 + \epsilon \ \cdots \ e_{k-P+1}^2 + \epsilon] \in \mathcal{R}^{1\times P}$ and ϵ is a very small positive scalar of the order of 10^{-12} whereas $\lambda = 1 - 1/(c_1 M)$ is the forgetting factor and c_1 is an integer. Eqs. (2.8)–(2.10) yield error bound γ_k which is then used in (2.5) to obtain α_k which is then used in (2.4) to obtain the weight vector $\boldsymbol{w}_k$.

2.2.2 RSMAP2 Algorithm

The RSMAP2 algorithm uses all of the equations of the RSMAP1 algorithm, i.e., (2.8) to (2.10), along with

$$\begin{aligned} \eta_k &= \beta\eta_{k-1} + (1-\beta)\text{min}\left(\eta_{k-1}, \frac{|d_k^2 - y_k^2|}{d_k^2}\right) & (2.11) \\ \sigma_{2,k}^2 &= \lambda\sigma_{2,k-1}^2 + (1-\lambda)\text{min}(\sigma_{2,k-1}^2, \sigma_{1,k}^2) & (2.12) \end{aligned}$$

where η_k is a parameter used to control the switching between error bounds γ_c and $\|\boldsymbol{e}_k\|_\infty - \nu\theta_k$, $\sigma_{2,k}^2$ is the variance of the error signal, $\beta = 1 - 1/(c_2 M)$ is the forgetting factor used to obtain the threshold

$$\gamma_c^2 = \gamma_{c,0}^2 + \Upsilon[1 + \text{sign}(1-\eta_k)]\sigma_{2,k}^2 \qquad (2.13)$$

where Υ is a tuning constant and $\gamma_{c,0}^2$ is an approximate estimate of the noise variance, σ_v^2. If no information is available about the noise variance, a zero value can be assigned to $\gamma_{c,0}^2$. In short, the RSMAP2 algorithm uses Eqs. (2.9)–(2.10) to obtain $\sigma_{1,k}$ and, in turn, evaluates $\theta_k = 1.88\sigma_{1,k}$; it then uses Eqs. (2.11)–(2.13) to obtain γ_c and (2.8) to obtain γ. With γ known, Eq. (2.5) yields α_k which is then used in (2.4) to obtain the new weight vector $\boldsymbol{w}_k$. The estimate γ_c in (2.13) is different from that used in the SMLMS family in [91], [92]. Several algorithms of the LMS and least-squares families that use the Hampel and Huber functions [66] to achieve robust performance with respect to outliers have been proposed in [93, 94, 63, 64].

2.2.3 Discussion

We have used the median of the error-squared signal samples over a finite window for the evaluation of $\sigma_{1,k}$ in (2.10) in order to render the estimate of $\sigma_{1,k}$ robust with respect to outliers for both the RSMAP1 and RSMAP2 algorithms. On the other hand, $\sigma_{2,k}$ is used to estimate the error bound γ_c for the RSMAP2 algorithm. In order to achieve robustness with respect to sudden system changes the estimator of $\sigma_{2,k}$ uses the minimum of the previous and current values of $\sigma_{1,k}$. Eq. (2.11) controls how long γ_c in (2.13) would work with $\gamma_{c,0}$.

As reported in [39], with a small γ_k the SSMAP algorithm yields faster convergence and increased steady-state misalignment; on the other hand, with a large γ it yields slower convergence and reduced steady-sate misalignment. Under these circumstances, optimal performance can be achieved by using a small error bound during transience and a large error bound during steady state. In the proposed RSMAP algorithm, optimal performance is achieved by choosing the initial values of $\sigma_{1,k}$, $\sigma_{2,k}$, and η_k to be large. In such a situation, during transience the algorithm would work with error bound $\gamma = \gamma_c$ which is significantly smaller than $\|\boldsymbol{e}_k\|_\infty - \nu\theta_k$ and as a result, the transient state would decay at a fast rate. On the other hand, during steady state the algorithm would work with error bound $\gamma = \|\boldsymbol{e}_k\|_\infty - \nu\theta_k$ as $\theta_k < \gamma_c$ during steady state which would yield reduced steady-state misalignment. In addition, this choice of γ_k would work in the occurrence of impulsive noise to yield robust performance with respect to outliers. In the presence of a sudden system disturbance, θ_k would tend to grow and, therefore, error bound $\gamma = \gamma_c$ would come into play. As a result, the re-adaptation capability of the proposed algorithm would be retained. Although a rough choice of the initial values of $\sigma_{1,k}$, $\sigma_{2,k}$, and η_k would work, simulation results have shown that the choices $\sigma_{1,0} = 20E_1/\hat{\sigma}_v^2$, $\sigma_{2,0} = 20E_2/\hat{\sigma}_v^2$, and $\eta_0 = 20E_3/\hat{\sigma}_v^2$, where $\hat{\sigma}_v^2$ is a crude approximation of the noise variance, give good results. Parameters E_1, E_2, E_3, c_1, c_2 are integers which can be chosen in the range 1 to 8 based on simulation results. The proposed algorithms are more robust with respect to the choice of tuning parameters for medium to high signal-to-noise ratios (SNRs) as compared to low SNRs. Parameter Υ in the RSMAP2 algorithm is a constant chosen to be 2.5. The regularization matrix $\delta\boldsymbol{I}$, where $\boldsymbol{I}$ is the $L \times L$ identity matrix and δ is a small constant, is added to the correlation matrix $\boldsymbol{X}_k^T\boldsymbol{X}_k$ in (2.4) and (2.6), and (2.7) to assure its invertibility if the signal becomes ill-conditioned. The value of δ should be very small of the order of 10^{-6} so that it does not noticeably influence the behavior of the algorithm.

The steady-state performance of the proposed algorithm is not influenced by the tuning parameters as it is determined by parameter ν. Improved convergence speed can be achieved by tuning the algorithm parameters as suggested. The convergence speed could become sensitive to changes in the tuning parameters especially for low SNRs and unknown systems of high order. However, no such problems have been experienced for high SNRs and unknown systems of low to moderate orders. For unknown systems of low order, say, 7 to 15, of the type commonly used in communication systems, the tuning parameters are easy to adjust and the performance of the adaptive filter is quite robust even for relatively low SNRs.

In the RSMAP1 algorithm, σ_v is prespecified and hence it can be chosen to achieve a good balance between convergence speed and computational savings. On the other hand, the RSMAP2 algorithm can be used in applications where σ_v cannot be prespecified. In most other situations, the two versions of the proposed algorithm offer similar performance.

2.3 Steady-State Analysis of RSMAP Algorithm

In this section, we provide a steady-state analysis of the proposed RSMAP algorithm. The analysis is based on the framework of an energy-conservation relation described in [20], [87] which was used to analyze several adaptation algorithms, for example, in [95], [96], [97]. Since the formula in (2.4) is essentially the same as that in (2.7), the steady-state analysis of the proposed RSMAP algorithm can be carried out by replacing μ in (2.7) by α_k given by (2.5) and then proceeding as in [20]. The update equation in (2.4) can be expressed in terms of the weight-error vector $\hat{\boldsymbol{w}}_k = \boldsymbol{w}_{opt} - \boldsymbol{w}_k$ as

$$\hat{\boldsymbol{w}}_k = \hat{\boldsymbol{w}}_{k-1} - \alpha_k \boldsymbol{X}_k (\delta \boldsymbol{I} + \boldsymbol{X}_k^T \boldsymbol{X}_k)^{-1} \boldsymbol{e}_k \tag{2.14}$$

where $\boldsymbol{w}_{opt}$ denotes the weight vector of the unknown system and $\delta \boldsymbol{I}$ is the regularization matrix. Premultiplying both sides in (2.14) by the input signal matrix, we obtain

$$\boldsymbol{X}_k^T \hat{\boldsymbol{w}}_k = \boldsymbol{X}_k^T \hat{\boldsymbol{w}}_{k-1} - \alpha_k \boldsymbol{X}_k^T \boldsymbol{X}_k (\delta \boldsymbol{I} + \boldsymbol{X}_k^T \boldsymbol{X}_k)^{-1} \boldsymbol{e}_k \tag{2.15}$$

Using the definition of the noise-free *a posteriori* error $\boldsymbol{\epsilon}_{f,k}$ given by

$$\boldsymbol{\epsilon}_{f,k} = \boldsymbol{X}_k^T \hat{\boldsymbol{w}}_k \tag{2.16}$$

and the noise-free *a priori* error $\boldsymbol{e}_{f,k}$ given by

$$\hat{\boldsymbol{e}}_{f,k} = \boldsymbol{X}_k^T \hat{\boldsymbol{w}}_{k-1} \tag{2.17}$$

in (2.15), we obtain

$$\boldsymbol{\epsilon}_{f,k} = \boldsymbol{e}_{f,k} - \alpha_k \boldsymbol{X}_k^T \boldsymbol{X}_k (\delta \boldsymbol{I} + \boldsymbol{X}_k^T \boldsymbol{X}_k)^{-1} \boldsymbol{e}_k \tag{2.18}$$

Hence, we can write

$$\alpha_k (\delta \boldsymbol{I} + \boldsymbol{X}_k^T \boldsymbol{X}_k)^{-1} \boldsymbol{e}_k = \left(\boldsymbol{X}_k^T \boldsymbol{X}_k\right)^{-1} (\boldsymbol{e}_{f,k} - \boldsymbol{\epsilon}_{f,k}) \tag{2.19}$$

Substituting (2.19) into (2.14), we get

$$\hat{\boldsymbol{w}}_k + \boldsymbol{X}_k \left(\boldsymbol{X}_k^T \boldsymbol{X}_k\right)^{-1} \boldsymbol{e}_{f,k} = \hat{\boldsymbol{w}}_{k-1} + \boldsymbol{X}_k \left(\boldsymbol{X}_k^T \boldsymbol{X}_k\right)^{-1} \boldsymbol{\epsilon}_{f,k} \tag{2.20}$$

By taking the square of the L_2 norm on both sides of the above equation after some manipulation, we obtain the following energy conservation relation [20]

$$\|\hat{\boldsymbol{w}}_k\|^2 + \boldsymbol{e}_{f,k}^T \left(\boldsymbol{X}_k^T \boldsymbol{X}_k\right)^{-1} \boldsymbol{e}_{f,k} = \|\hat{\boldsymbol{w}}_{k-1}\|^2 + \boldsymbol{\epsilon}_{f,k}^T \left(\boldsymbol{X}_k^T \boldsymbol{X}_k\right)^{-1} \boldsymbol{\epsilon}_{f,k} \tag{2.21}$$

Taking the expectation on both sides of (2.21), we obtain

$$\begin{aligned} E\left[\|\hat{\boldsymbol{w}}_k\|^2\right] + E\left[\boldsymbol{e}_{f,k}^T \left(\boldsymbol{X}_k^T \boldsymbol{X}_k\right)^{-1} \boldsymbol{e}_{f,k}\right] =& E\left[\|\hat{\boldsymbol{w}}_{k-1}\|^2\right] + \\ & E\left[\boldsymbol{\epsilon}_{f,k}^T \left(\boldsymbol{X}_k^T \boldsymbol{X}_k\right)^{-1} \boldsymbol{\epsilon}_{f,k}\right] \end{aligned} \tag{2.22}$$

At steady state, we have $E\left[\|\hat{\boldsymbol{w}}_k\|^2\right] = E\left[\|\hat{\boldsymbol{w}}_{k-1}\|^2\right]$ and hence the above relation assumes the form

$$E\left[\boldsymbol{e}_{f,k}^T \left(\boldsymbol{X}_k^T \boldsymbol{X}_k\right)^{-1} \boldsymbol{e}_{f,k}\right] = E\left[\boldsymbol{\epsilon}_{f,k}^T \left(\boldsymbol{X}_k^T \boldsymbol{X}_k\right)^{-1} \boldsymbol{\epsilon}_{f,k}\right] \tag{2.23}$$

Substituting (2.18) into (2.23), we obtain

$$E\left[\boldsymbol{e}_{f,k}^T\left(\boldsymbol{X}_k^T\boldsymbol{X}_k\right)^{-1}\boldsymbol{e}_{f,k}\right] = E\Big[\boldsymbol{e}_{f,k}^T\left(\boldsymbol{X}_k^T\boldsymbol{X}_k\right)^{-1}\boldsymbol{e}_{f,k}^T -\alpha_k\boldsymbol{e}_{f,k}^T\left(\delta\boldsymbol{I}+\boldsymbol{X}_k^T\boldsymbol{X}_k\right)^{-1}\boldsymbol{e}_k - \alpha_k\boldsymbol{e}_k^T\left(\delta\boldsymbol{I}+\boldsymbol{X}_k^T\boldsymbol{X}_k\right)^{-1}\boldsymbol{e}_{f,k} +\alpha_k^2\boldsymbol{e}_k^T\left(\delta\boldsymbol{I}+\boldsymbol{X}_k^T\boldsymbol{X}_k\right)^{-1}\left(\boldsymbol{X}_k^T\boldsymbol{X}_k\right)\left(\delta\boldsymbol{I}+\boldsymbol{X}_k^T\boldsymbol{X}_k\right)^{-1}\boldsymbol{e}_k\Big] \tag{2.24}$$

Using the definition of the input-signal correlation matrix, i.e., $\boldsymbol{R}_k = \boldsymbol{X}_k^T\boldsymbol{X}_k$, and assuming that $\boldsymbol{S}_k = \left(\delta\boldsymbol{I}+\boldsymbol{X}_k^T\boldsymbol{X}_k\right)^{-1} \approx \boldsymbol{R}_k^{-1}$ for $\delta \ll 1$, (2.24) can be simplified to

$$E\left[\alpha_k^2\boldsymbol{e}_k^T\boldsymbol{S}_k\boldsymbol{e}_k\right] = E\left[\alpha_k\boldsymbol{e}_{f,k}^T\boldsymbol{S}_k\boldsymbol{e}_k\right] + E\left[\alpha_k\boldsymbol{e}_k^T\boldsymbol{S}_k\boldsymbol{e}_{f,k}\right] \tag{2.25}$$

At steady state $\|\boldsymbol{e}_k\|_\infty \approx |e_k|$ and $|e_k|/\sigma_{1,k} \to 1$. Since $\gamma_k = \|\boldsymbol{e}_k\|_\infty - \nu\theta_k$ at steady state, then for a small value of ν (2.5) yields

$$\alpha_k \approx 1.88\nu \tag{2.26}$$

Thus by using a deterministic value for α_k, (2.25) simplifies to

$$\alpha_k E\left[\boldsymbol{e}_k^T\boldsymbol{S}_k\boldsymbol{e}_k\right] = E\left[\boldsymbol{e}_{f,k}^T\boldsymbol{S}_k\boldsymbol{e}_k\right] + E\left[\boldsymbol{e}_k^T\boldsymbol{S}_k\boldsymbol{e}_{f,k}\right] \tag{2.27}$$

Now substituting

$$\boldsymbol{e}_k = \boldsymbol{e}_{f,k} + \boldsymbol{v}_k \tag{2.28}$$

into (2.27), we obtain

$$\alpha_k E\left[(\boldsymbol{e}_{f,k}+\boldsymbol{v}_k)^T\boldsymbol{S}_k(\boldsymbol{e}_{f,k}+\boldsymbol{v}_k)\right] = 2E\left[\boldsymbol{e}_{f,k}^T\boldsymbol{S}_k(\boldsymbol{e}_{f,k}+\boldsymbol{v}_k)\right] \tag{2.29}$$

Considering the noise signal $\boldsymbol{v}_k$ to be white as well as statistically independent of the input signal and neglecting the dependency of $\boldsymbol{e}_{f,k}$ on $\boldsymbol{v}_k$, the expression in (2.29) can be further simplified to

$$\alpha_k E\left[\boldsymbol{e}_{f,k}^T\boldsymbol{S}_k\boldsymbol{e}_{f,k}\right] + \alpha_k E\left[\boldsymbol{v}_k^T\boldsymbol{S}_k\boldsymbol{v}_k\right] = 2E\left[\boldsymbol{e}_{f,k}^T\boldsymbol{S}_k\boldsymbol{e}_{f,k}\right] \tag{2.30}$$

Applying the trace operation to both sides of (2.30), we obtain

$$E\left\{\mathrm{tr}\left[\boldsymbol{e}_{f,k}\boldsymbol{e}_{f,k}^T\boldsymbol{S}_k\right]\right\} = \frac{\alpha_k}{2-\alpha_k}E\left\{\mathrm{tr}\left[\boldsymbol{v}_k\boldsymbol{v}_k^T\boldsymbol{S}_k\right]\right\} \tag{2.31}$$

Now assuming that $\boldsymbol{S}_k$ is statistically independent of both the *a priori* error at steady state and the noise signal, we obtain

$$\text{tr}\Big\{E\left[\boldsymbol{e}_{f,k}\boldsymbol{e}_{f,k}^T\right]E\left[\boldsymbol{S}_k\right]\Big\} = \frac{\alpha_k}{2-\alpha_k}\text{tr}\left\{E\left[\boldsymbol{v}_k\boldsymbol{v}_k^T\right]E\left[\boldsymbol{S}_k\right]\right\} \tag{2.32}$$

This result is used in the next subsection to derive an expression for the excess MSE in the RSMAP algorithm.

2.3.1 Excess MSE in RSMAP Algorithm

In order to derive an expression for the excess MSE in the proposed RSMAP algorithm, we have to deduce an expression for the noise-free error covariance $E\left[\boldsymbol{e}_{f,k}\boldsymbol{e}_{f,k}^T\right]$. The noise-free *a posteriori* error $\boldsymbol{\epsilon}_{f,k}$ in (2.16) can also be expressed as

$$\boldsymbol{\epsilon}_{f,k} = \boldsymbol{e}_{f,k} - \alpha_k\boldsymbol{e}_k = (1-\alpha_k)\boldsymbol{e}_{f,k} - \alpha_k\boldsymbol{v}_k \tag{2.33}$$

Now following the steps in [20], we obtain

$$E\left[\boldsymbol{e}_{f,k}\boldsymbol{e}_{f,k}^T\right] = E\left[e_{f,k}^2\right]\boldsymbol{U}_1 + E[\alpha_k^2]\sigma_v^2\boldsymbol{U}_2 \tag{2.34}$$

where $\boldsymbol{U}_1$ and $\boldsymbol{U}_2$ are diagonal matrices given by

$$\boldsymbol{U}_1 = \begin{bmatrix} 1 & & & \\ & (1-\alpha_k)^2 & & \\ & & \ddots & \\ & & & (1-\alpha_k)^{2(L-1)} \end{bmatrix} \tag{2.35}$$

and

$$\boldsymbol{U}_2 = \begin{bmatrix} 0 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1+\cdots+(1-\alpha_k)^{2L} \end{bmatrix} \tag{2.36}$$

respectively. Since the step size α_k is very small at steady state, we obtain $\alpha_k^2\sigma_v^2\boldsymbol{U}_2 \approx \boldsymbol{0}$ and $\boldsymbol{U}_1 \approx \boldsymbol{I}$. Consequently, (2.34) assumes the form

$$E\left[\boldsymbol{e}_{f,k}\boldsymbol{e}_{f,k}^T\right] = E\left[e_{f,k}^2\right]\boldsymbol{U}_1 \tag{2.37}$$

Using (2.37) in (2.32), we obtain

$$E\left[e_{f,k}^2\right] = \frac{\alpha_k}{2-\alpha_k}\sigma_v^2 \frac{\operatorname{tr}\left\{E\left[\boldsymbol{S}_k\right]\right\}}{\operatorname{tr}\left\{\boldsymbol{U}_1 E\left[\boldsymbol{S}_k\right]\right\}} \tag{2.38}$$

For a sufficiently small α_k, we can assume that $\boldsymbol{U}_1 \approx \boldsymbol{I}$ and in this case (2.38) yields the excess MSE as

$$\text{EMSE} = \frac{\alpha_k}{2-\alpha_k}\sigma_v^2 \tag{2.39}$$

As can be seen, the excess MSE is independent of the projection order.

2.3.2 Verification of EMSE

In this subsection, we examine the accuracy of the expression for the EMSE given in (2.39) in a system-identification application. The unknown system in this experiment was a 16th-order finite-duration impulse response (FIR) filter with a cut-off frequency of 0.3, which was designed using MATLAB command $\boldsymbol{w}_{opt} = fir1(M-1, 0.3)$. The weight vector of the unknown system was normalized to have unity power. An infinite-duration impulse response (IIR) filter with a pole at 0.95 was used to produce a colored input signal from a zero-mean white Gaussian noise signal with a variance of unity. The desired signal was contaminated with a zero-mean white Gaussian noise signal with variance $\sigma_v^2 = 10^{-4}$. The expected steady-state MSE was obtained as $MSE = \sigma_v^2 + EMSE$ where EMSE is given in Eqn. (2.39). The parameters for the proposed RSMAP1 algorithm were set to $c_1 = 1$, $E_1 = 1$, $\boldsymbol{w}_0 = \boldsymbol{0}$, $\gamma_c = \sqrt{5\sigma_v^2}$ for all L. The parameters for the proposed RSMAP2 algorithm were set to $\kappa = 2.5$, $\boldsymbol{w}_0 = \boldsymbol{0}$, $E_2 = E_3 = 1$ for all L and $c_2 = 2$ and 3 for $L = [2, 4]$ and $L = [6, 8]$, respectively. The relative error between the expected and actual MSE defined as

$$\text{Relative error} = \frac{\text{Expected MSE} - \text{Actual MSE}}{\text{Expected MSE}} 100$$

with fixed and variable thresholds for different values of L and ν is plotted in Figs. 2.1–2.2. As can be seen, the formula in (2.39) provides a fairly accurate estimate of the expected MSE for values of ν in the range $\nu = (0, 0.5]$. It should be mentioned that a reduced ν does not affect the convergence speed or the tracking capability and at the same time for values of ν in the range $(0, 0.2]$, the steady-state MSE does not increase noticeably as L is increased as can be seen in Figs. 2.1–2.2.

The above analysis can be used to select appropriate values for parameters ν and

L.

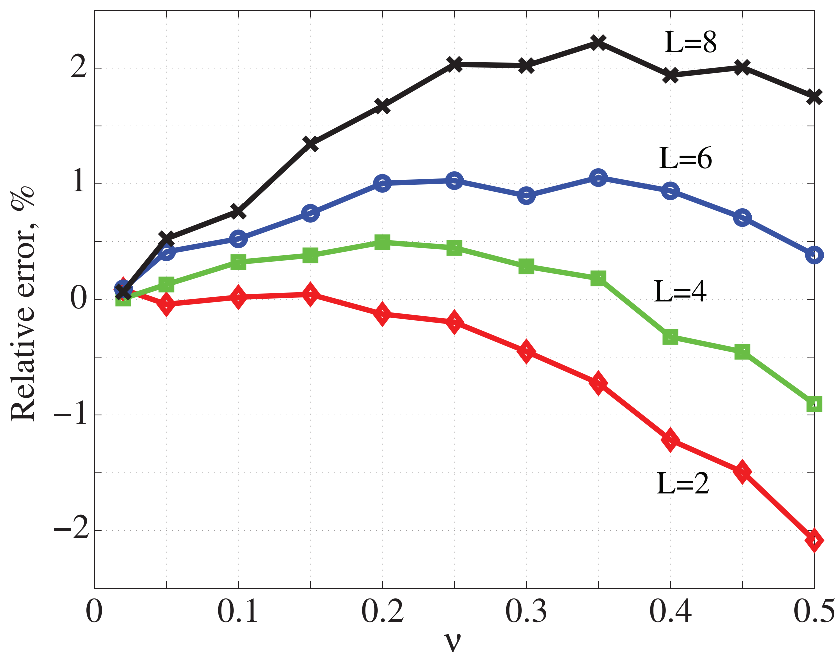

Figure 2.1: Relative error between expected and experimental MSE: RSMAP algorithm with fixed threshold.

2.4 Simulation Results

In this section, we compare the performance of the two versions of the RSMAP algorithm with that of some known AP and SMAP algorithms through extensive simulations. Two applications, namely, a system-identification application and an acoustic-echo-cancelation application, were considered.

2.4.1 System Identification Application

In the first set of experimental results, the performance of the RSMAP1 and RSMAP2 algorithms is compared with that of the conventional AP, SSMAP, and SM proportionate AP (SMPAP) algorithms reported in [11], [19], and [25], respectively, for a system-identification application. A network echo path of length $M = 96$ with the impulse response illustrated in Fig. 2.3 was chosen as the unknown system. A col-

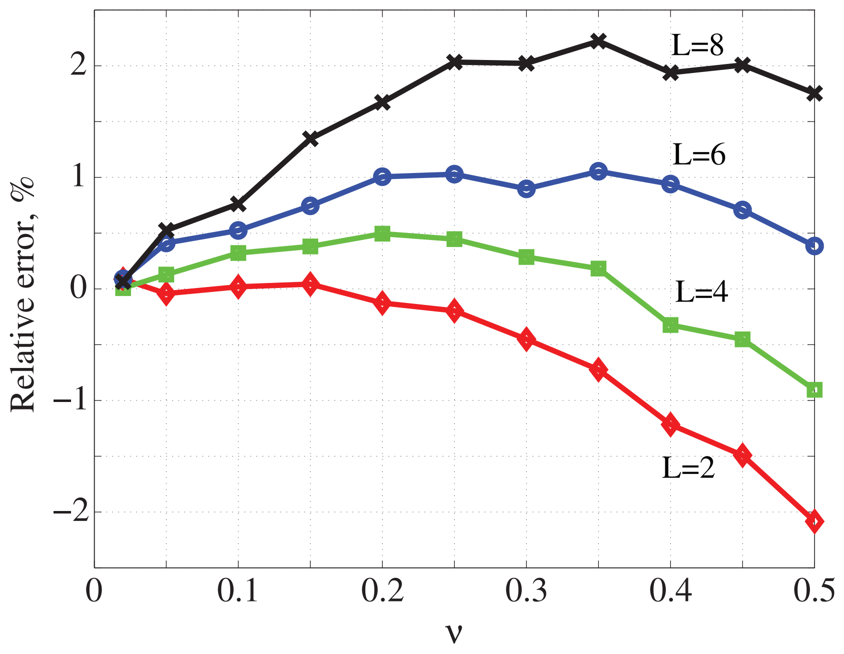

Figure 2.2: Relative error between expected and experimental MSE: RSMAP algorithm with variable threshold.

ored input signal for the adaptive filter was generated by filtering a zero-mean white Gaussian noise signal with power 10 by using an IIR filter with transfer function [98]

$$H(z) = \frac{1 + 0.5z^{-1} + 0.81z^{-2}}{1 - 0.59z^{-1} + 0.4z^{-2}} \tag{2.40}$$

The eigenvalue spread ratio of the colored input signal of 10000 samples was obtained as 462 by using the ensemble average of the autocorrelation matrix given by (1.28) over 1000 independent trials with $\lambda = 1 - 2^{-15}$ and $\boldsymbol{R}_0 = 10^{-4}\boldsymbol{I}$. The impulsive noise was modeled as $\phi_k = \omega_k \chi_k$ where ω_k is a Bernoulli process with $P(\omega_k = 1) = p = 0.01$ and χ_k is a zero-mean Gaussian signal with variance $\sigma_\chi^2 = 10^4 \sigma_y^2$ where σ_y^2 is the power of the uncorrupted output signal [68]. The weight vector was initialized as $\boldsymbol{w} = \boldsymbol{0}$ in all algorithms and all experiments. In addition, the error bound, γ for the SSMAP and SMPAP algorithms and γ_c for the RSMAP1 algorithm were set to $\sqrt{5\sigma_v^2}$ in all experiments. Unless otherwise stated, the parameters of the RSMAP algorithms were chosen as $c_1 = 1$, $P = 15$, $\nu = 0.05$, $c_2 = 1$, $E_1 = 2$, $E_2 = 2$, and $E_3 = 2$.

The measurement noise, v_k, added to the desired signal was also a Gaussian noise signal with a zero mean and variance $\sigma_v^2 = 10^{-3}$. The SNR was 30 dB. The impulse

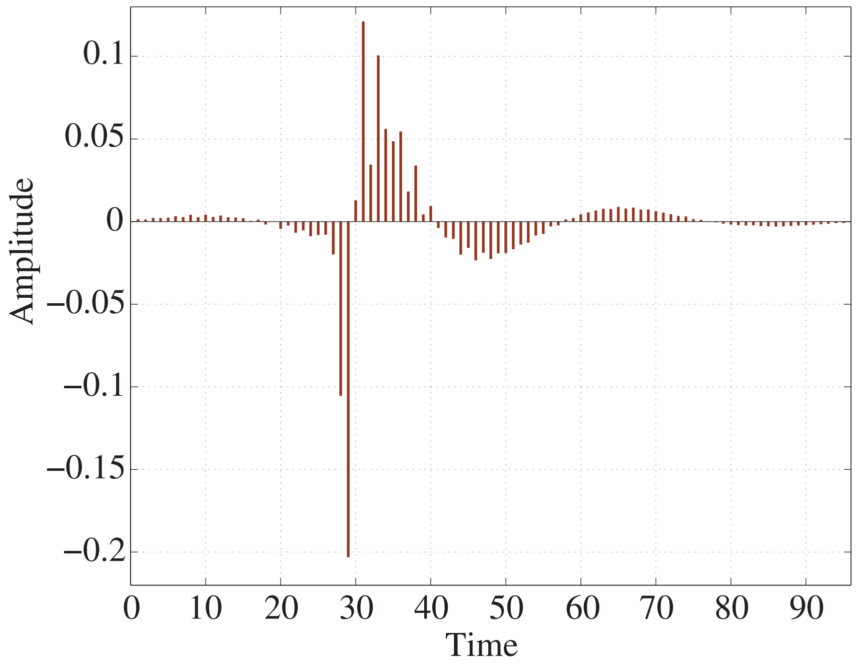

Figure 2.3: Impulse response of a network echo path.

response of the echo path was multiplied by -1 at iteration 10^4 to investigate the re-adaptation capability of the algorithms. Impulsive noise was added to the desired signal at iterations 5000 and $15,000$. The learning curves obtained averaged over 1000 independent trials for $L = 8$ are illustrated in Fig. 2.4. As can be seen, the proposed RSMAP algorithms yield significantly reduced steady-state misalignment and improved robustness with respect to impulsive noise for the same initial convergence speed when compared to the other algorithms. The evolution of the averaged step size α_k for the SSMAP, SMPAP, and the proposed RSMAP algorithms is illustrated in Fig. 2.5. As can be seen, the proposed algorithms yield lower values of α_k during steady state, namely, 0.004, compared to 0.0834 and 0.0823 in the SSMAP and SMPAP algorithms, respectively. As a result, a much reduced steady-state misalignment is achieved in the proposed algorithms. In addition, since α_k returns back to high values during sudden system changes, the proposed algorithms preserve their re-adaptation capability.

To examine the effect of projection order on the performance of the proposed algorithms, we repeated the same experiment with the variance of the measurement noise changed to $\sigma_v^2 = 10^{-6}$ which corresponds to a SNR of 60 dB for values of

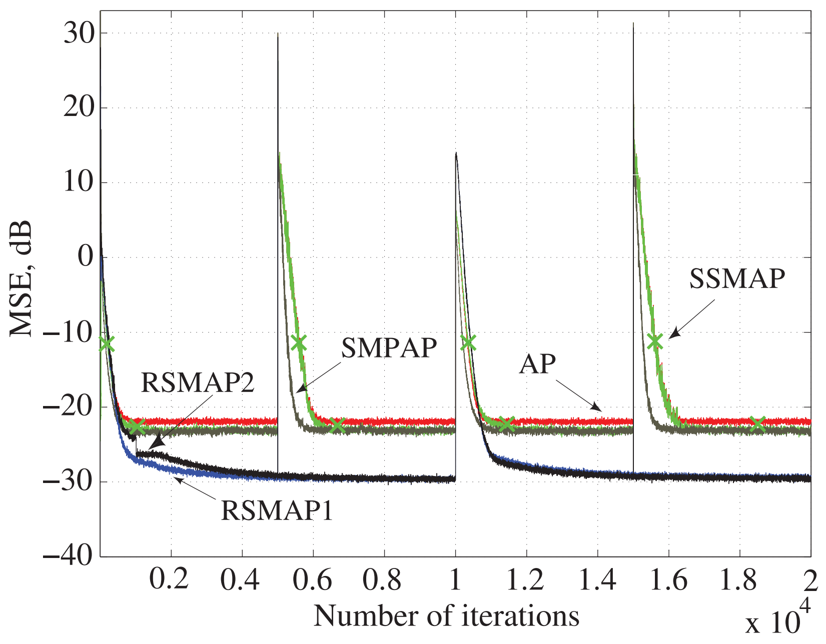

Figure 2.4: Learning curves with $L = 8$, in all algorithms.

$L = 2,\ 4,\ 6$, and 8. In this experiment, the parameters of the RSMAP algorithms were chosen as $E_1 = E_2 = E_3 = 1$ and impulsive noise was added to the desired signal at iteration $25,000$. Parameter c_1 was set to 6, 3, 2, and 1 to obtain the plots in Figs. 2.6–2.9, respectively, and c_2 was chosen to be equal to c_1. The learning curves obtained in 1000 independent trials are illustrated in Figs. 2.6–2.9. As can be seen, as the projection order is increased, the convergence speed of the AP and SSMAP algorithms is increased at the cost of increased steady-state misalignment which is consistent with the analysis presented in [20] and [39]. On the other hand, the proposed algorithms yield significantly reduced steady-state misalignment relative to that in the AP and SSMAP algorithms especially when the projection order is large. The numerical values of the steady-state MSE in dB for the AP, SSMAP, RSMAP1, and RSMAP2 algorithms for $L = 2$ and $L = 8$ are given in Table 2.1. As can be seen, the use of the proposed algorithms leads to a reduction in the steady-state misalignment in the range of 4.0 to 7.9 dB relative to the AP algorithm and 0.4 to 6.8 dB relative the SSMAP algorithm depending on the value of L. In addition, the proposed RSMAP algorithms offer improved robustness with respect to impulsive

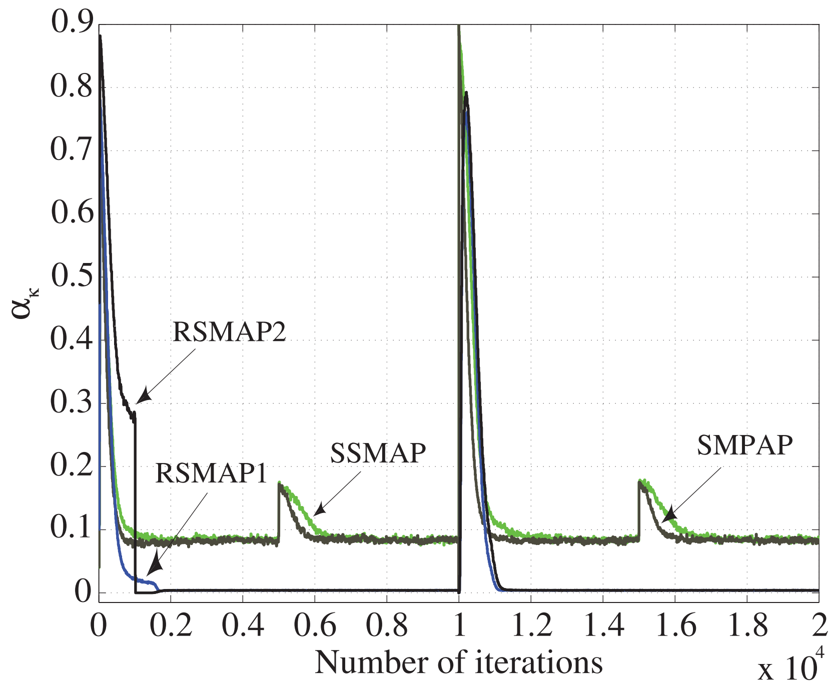

Figure 2.5: Evolution of α_k.

Table 2.1: Numerical Values of the Steady-State MSE, dB

L	AP	SSMAP	RSMAP1	RSMAP2
2	-54.6	-58.2	-58.6	-58.9
8	-51.9	-53.0	-59.8	-59.8

noise.

2.4.2 Acoustic-Echo-Cancelation Application

In the second set of experimental results, the performance of the RSMAP algorithms is compared with that of the competing algorithms for the case of an acoustic-echo-cancelation application. In such an application, the length of the acoustic echo path is significantly larger than the length of the network echo path and the adaptive filter requires thousands of weights to successfully cancel the acoustic echo signal. The impulse response of the acoustic echo path assumed is illustrated in Fig. 2.10. Only the first 512 samples were used to produce the echo signal. An IIR filter with transfer function

$$H(z) = \frac{1}{z^4 - 0.95z^3 - 0.19z^2 - 0.09z + 0.5} \tag{2.41}$$

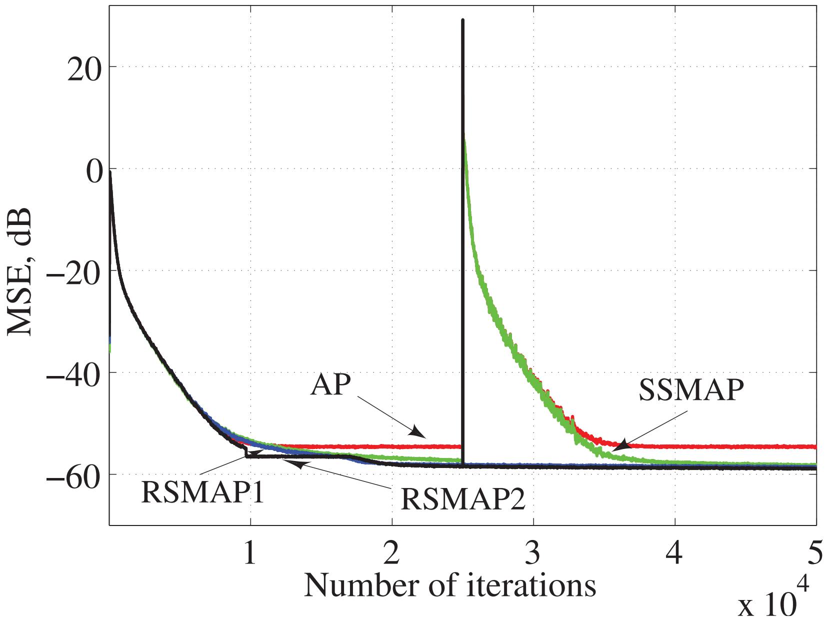

Figure 2.6: Learning curves with $L = 2$ in all algorithms.

was used to bandlimit a zero-mean white Gaussian noise signal with variance one that served as the input signal for the adaptive filters. The input signal would thus be more correlated than that used in the system-identification application. The eigenvalue spread ratio of this colored input signal of 10000 samples was obtained as 8971 by using the ensemble average of the autocorrelation matrix given by (1.28) over 1000 independent trials with $\lambda = 1 - 2^{-15}$ and $\boldsymbol{R}_0 = 10^{-4}\boldsymbol{I}$. The measurement noise was a zero-mean white Gaussian noise signal with variance 10^{-2} with impulsive noise at iterations $25,000$ and $75,000$. The impulse response of the echo path was multiplied by -1 at iteration $50,000$. The parameters for the proposed RSMAP algorithms were set to $E_1 = E_2 = E_3 = 5$ and $L = 8$ in all subsequent tests. The learning curves obtained by using the AP, SSMAP, SMPAP, RSMAP1, and RSMAP2 algorithms for 10^3 independent trials are illustrated in Fig. 2.11. As can be seen, the RSMAP algorithms yield a significant reduction in the steady-state misalignment for the same convergence speed compared to the other algorithms while retaining their re-adaptation capability. The proposed algorithms also exhibit robust performance with respect to impulsive noise.

Next we repeated the same experiment with real speech signals as the input signals. Twenty speech signals were recorded and each was then used as the input signal in a

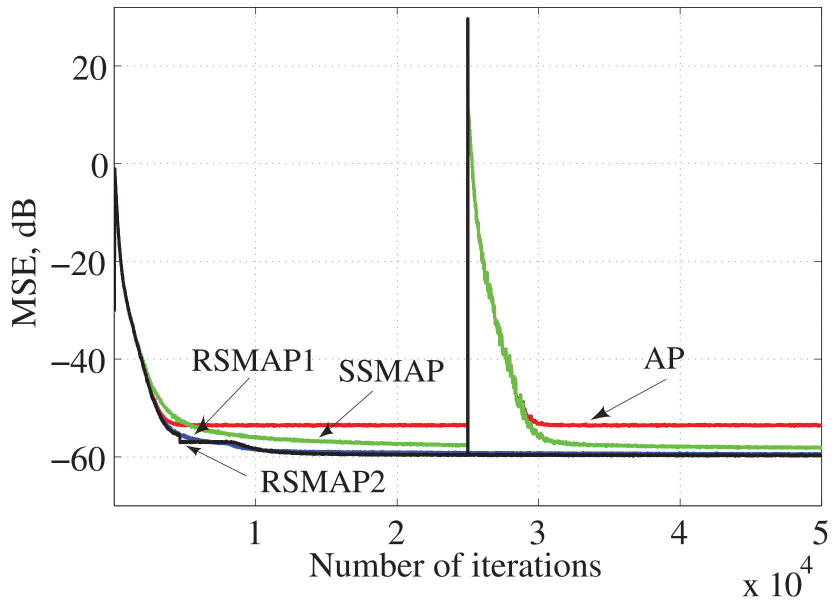

Figure 2.7: Learning curves with $L = 4$, in all algorithms.

given trial. The desired signal and each speech signal were contaminated with a zero-mean white Gaussian noise signal with a variance of 10^{-4}. In order to obtain a smooth learning curve, the misalignment was evaluated as $20 \log_{10} \left(\|\boldsymbol{w}_{opt} - \boldsymbol{w}_k\|_2 \,/\, \|\boldsymbol{w}_{opt}\|_2 \right)$ where $\boldsymbol{w}_{opt}$ is the impulse response shown in Fig. 2.10. The learning curves obtained in 20 independent trials are illustrated in Figs. 2.12–2.13. The plots of Fig. 2.12 correspond to the case where no impulsive noise was added to the desired signal. The plots of Fig. 2.13 correspond to the case where impulsive noise of duration $100T$ was added to the desired signal at iterations $50,000$ and $150,000$, where T is the sampling period, and the impulse response of the echo path was multiplied by -1 at iteration $100,000$. As can be seen in Figs. 2.12 and 2.13 the RSMAP1 and RSMAP2 algorithms yield reduced steady-state misalignment for the same convergence speed as compared to the other algorithms. In addition, the RSMAP1 and RSMAP2 algorithms are more robust with respect to a long burst of impulsive noise and offer similar re-adaptation relative to the other algorithms. The RSMAP2 algorithm yields slightly more misalignment compared to the RSMAP1 algorithm due to the fact that the latter algorithm starts with a much lower error bound than the former. The long burst of impulsive noise simulates cross-talk and intersymbol interference situations commonly found in two-way telephone and communication systems.

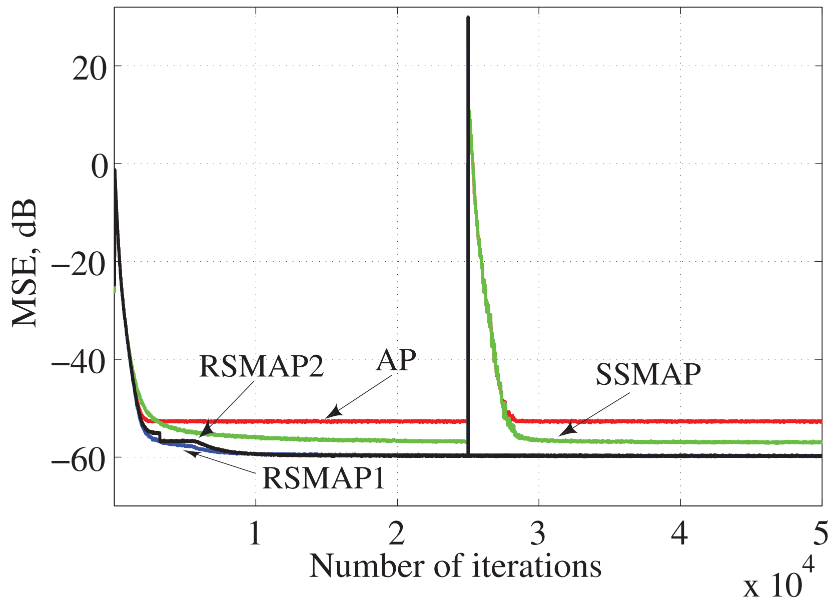

Figure 2.8: Learning curves with $L = 6$, in all algorithms.

2.5 Conclusions

A set-membership AP adaptation algorithm that achieves robustness through the use of two error bounds has been proposed. Through a steady-state analysis, an approximate formula for the expected MSE has been derived for the proposed algorithm, which gives relatively accurate values of the actual MSE. This formula can be used to estimate appropriate values for parameters ν and L. On the other hand, extensive simulation results have shown that the guidelines presented for the choice of parameters $\sigma_{1,0}$, $\sigma_{2,0}$, λ, β, η_0 work quite well in practice. The two versions of the proposed algorithm were applied to system-identification and acoustic-echo-cancelation applications. The simulation results obtained show that the RSMAP1 and RSMAP2 algorithms perform much better than the conventional AP, SSMAP, and SMPAP algorithms in terms of steady-state misalignment and robustness with respect to impulsive noise without compromising the initial convergence speed and re-adaptation capability.

In the next chapter we describe a constrained version of the proposed RSMAP algorithm.

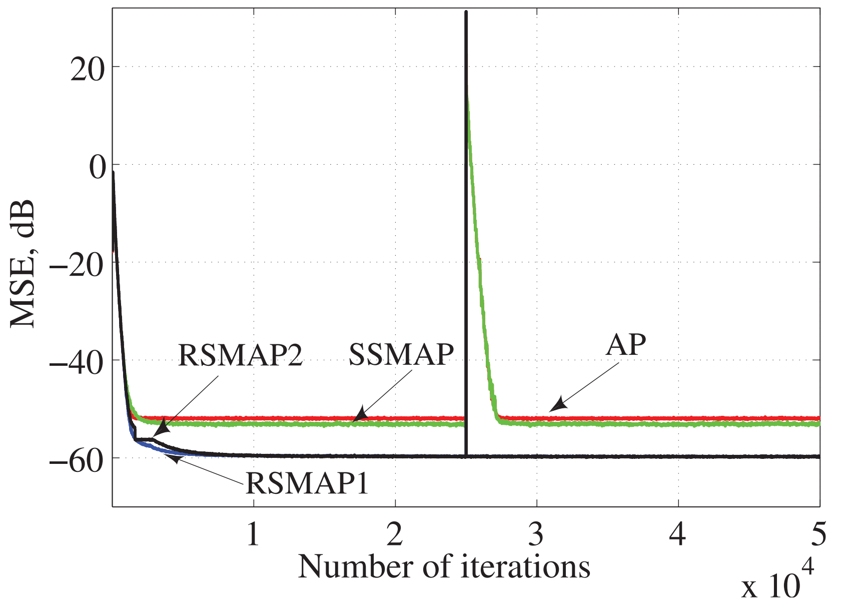

Figure 2.9: Learning curves with $L = 8$, in all algorithms.

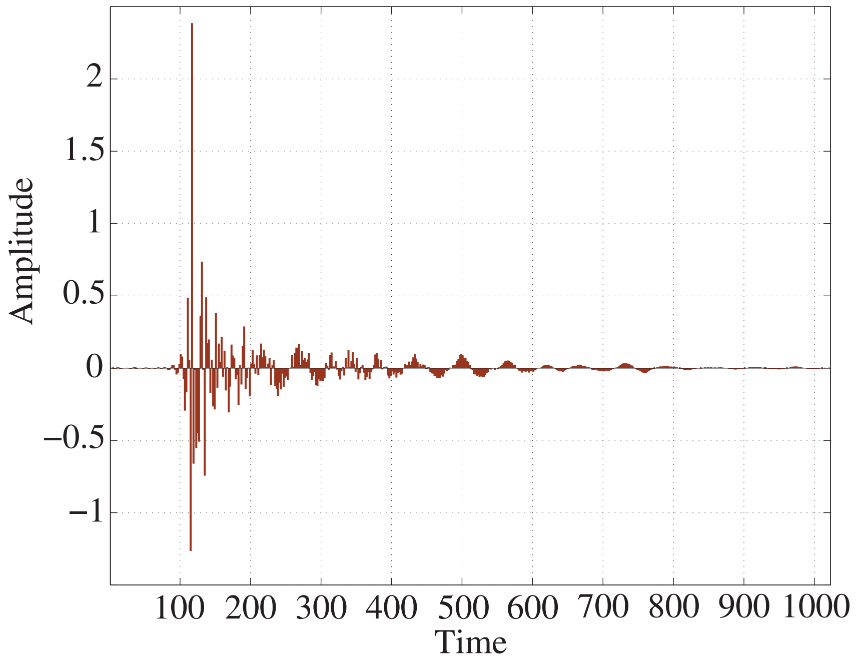

Figure 2.10: Impulse response of acoustic echo path.

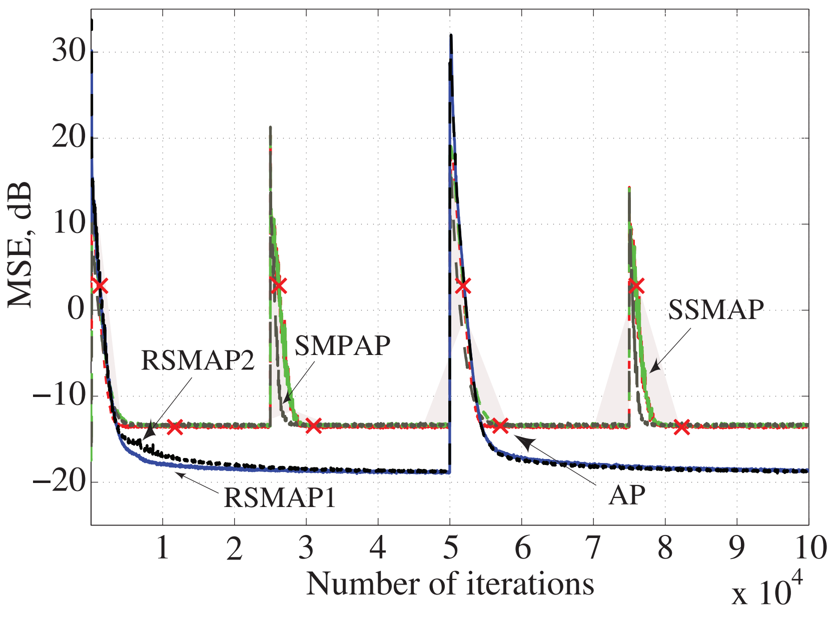

Figure 2.11: Learning curves with Gaussian input.

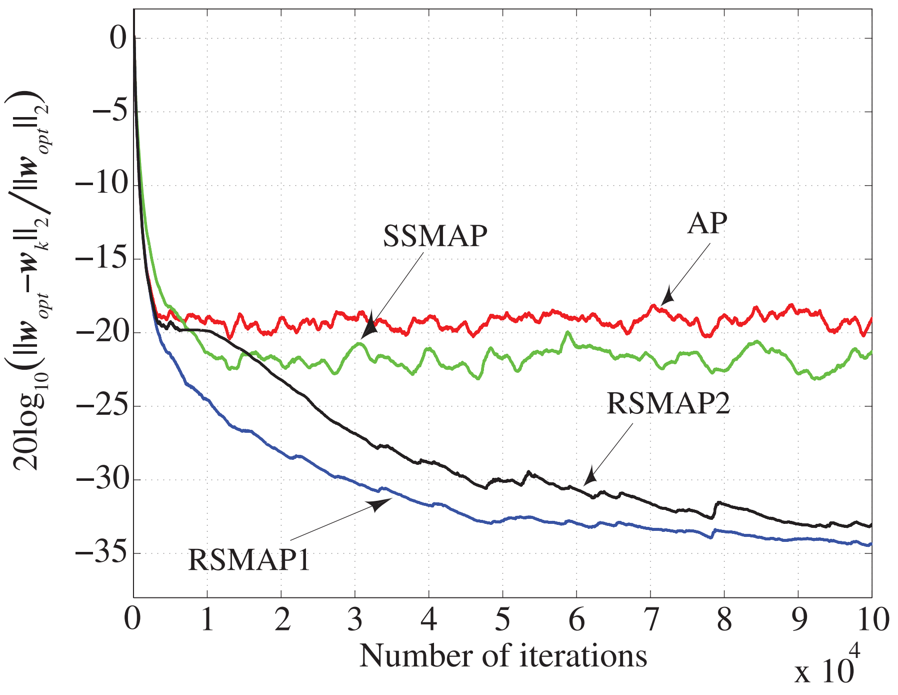

Figure 2.12: Simulation results for acoustic-echo cancelation application. Learning curves with speech signals as input.

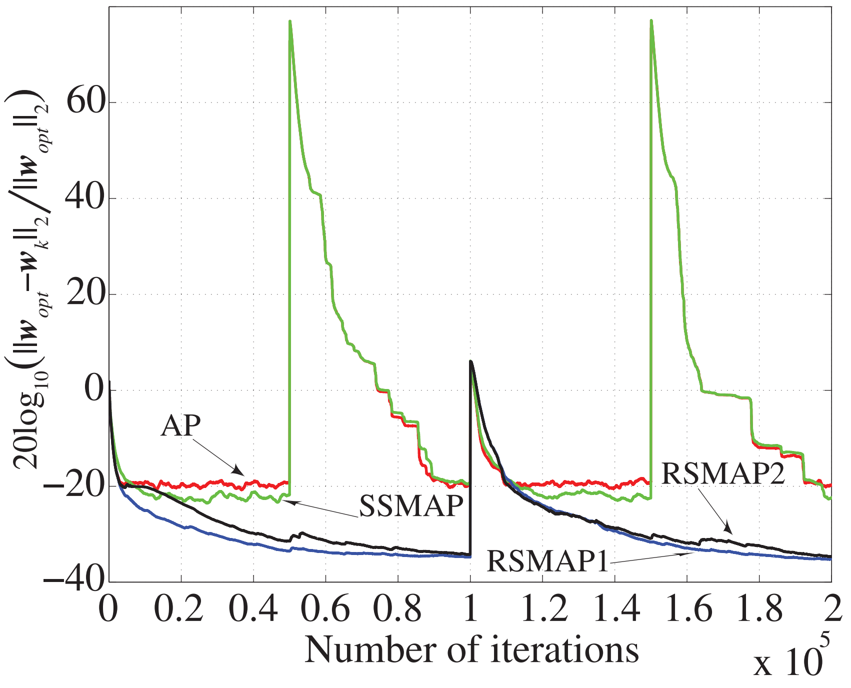

Figure 2.13: Simulation results for acoustic-echo cancelation application. Learning curves with $P = 215$, and speech signals as input.

Chapter 3

Robust Constrained Set-Membership Affine Projection Adaptation Algorithm

3.1 Introduction

We have seen in table 2.1 that the AP and SMAP adaptation algorithms yield increased misalignment for an increased projection order although a fast convergence can be achieved. Since the constrained affine projection (CAP) and constrained SM affine projection (CSMAP) algorithms reported in [27] are essentially constrained versions of the AP and SMAP algorithms, respectively, they also yield increased steady-state misalignment for an increased projection order. In addition, the performance of these algorithms is also affected by impulsive-noise interference.

In this chapter, a constrained robust SMAP is proposed [83], referred to hereafter as the PCSMAP algorithm, that solves the problem of increased steady-state misalignment brought about by an increased projection order and performs well in impulsive-noise environments. Like the RSMAP algorithm, the PCSMAP algorithm uses two error bounds. One of the error bounds yields faster convergence and good re-adaptation capability whereas the other reduces the influence of impulsive noise and yields a reduced steady-state misalignment without compromising the convergence speed. Switching between the two error bounds is achieved by comparing the absolute value of the error signal with a threshold. A simple variance estimator for the error signal is used to determine this threshold. Simulation results show that the proposed PCSMAP algorithm outperforms the CNLMS, CAP, and CSMAP algorithms in linear-phase plant identification, sinusoid filtering, and interference suppression in direct-sequence code division multiple access (DS-CDMA) communication systems applications.

3.2 Proposed Constrained SMAP Algorithm

The PCSMAP algorithm performs weight adaptation such that the updated weight vector belongs to the constraint set in (1.24) at the L most recent iterations, i.e., $\boldsymbol{w}_k \in \Psi_k^L$. Whenever the weight vector $\boldsymbol{w}_k$ is not a member of Ψ_k^L, an update is performed by solving the optimization problem [83]

$$\underset{\boldsymbol{w}_k}{\text{minimize}}\ J_{\boldsymbol{w}_k} = \|\boldsymbol{w}_k - \boldsymbol{w}_{k-1}\|^2 \tag{3.1}$$

subject to the constraints

$$\boldsymbol{C}\boldsymbol{w}_k = \boldsymbol{f} \tag{3.2}$$

$$\boldsymbol{d}_k - \boldsymbol{X}_k^T \boldsymbol{w}_k = \boldsymbol{g}_k \tag{3.3}$$

where $\boldsymbol{d}_k \in \mathcal{R}^{L\times 1}$ is the desired signal vector, $\boldsymbol{g}_k \in \mathcal{R}^{L\times 1}$ is the error-bound vector, $\boldsymbol{X}_k \in \mathcal{R}^{M\times L}$ is the input signal matrix, $\boldsymbol{C} \in \mathcal{R}^{N\times M}$ with $N < M$ is the constraint matrix, and $\boldsymbol{f} \in \mathcal{R}^{N\times 1}$ is the constraint vector. The error vector is obtained as $\boldsymbol{e}_k = [e_k\ \epsilon_{k-1} \cdots \epsilon_{k-L+1}]^T$ where $e_k = d_k - \boldsymbol{x}_k^T \boldsymbol{w}_{k-1}$ is the *a priori* error and $\epsilon_{k-i} = d_{k-i} - \boldsymbol{x}_{k-i}^T \boldsymbol{w}_{k-1}$ is the *a posteriori* error. We choose the error-bound vector as

$$\boldsymbol{g}_k = \gamma \left[\frac{e_k}{|e_k|}\ \frac{\epsilon_{k-1}}{|e_k|}\ \cdots\ \frac{\epsilon_{k-L+1}}{|e_k|} \right]^T \tag{3.4}$$

With this choice of the error bound vector, the solution of the problem in (3.1) yields the weight-vector update formula

$$\boldsymbol{w}_k = \boldsymbol{Z}\left[\boldsymbol{w}_k + \alpha_k \boldsymbol{X}_k (\boldsymbol{X}_k^T \boldsymbol{Z} \boldsymbol{X}_k)^{-1} \boldsymbol{e}_k\right] + \boldsymbol{F} \tag{3.5}$$

where

$$\alpha_k = \begin{cases} 1 - \dfrac{\gamma}{|e_k|} & \text{if } |e_k| > \gamma \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

and

$$\boldsymbol{Z} = \boldsymbol{I} - \boldsymbol{C}^T \left(\boldsymbol{C}\boldsymbol{C}^T\right)^{-1} \boldsymbol{C} \tag{3.7}$$

$$\boldsymbol{F} = \boldsymbol{C}^T \left(\boldsymbol{C}\boldsymbol{C}^T\right)^{-1} \boldsymbol{f} \tag{3.8}$$

Note that if $|e_k| > g_k(1)$, we obtain $|\epsilon_{k-i}| > g_k(i)$ for $i = 2, \ldots, L$ where $g_k(i)$ denotes the ith element of vector $\boldsymbol{g}_k$.

The error-bound vector in [27] is chosen as $\boldsymbol{g}_k = [\gamma \text{sign}(e_k) \ \epsilon_{k-1} \ \cdots \ \epsilon_{k-L+1}]^T$ which results in a weight-vector update formula

$$\boldsymbol{w}_k = \boldsymbol{Z} \left[\boldsymbol{w}_{k-1} + \alpha_k \boldsymbol{X}_k (\boldsymbol{X}_k^T \boldsymbol{Z} \boldsymbol{X}_k)^{-1} e_k \boldsymbol{u}_1\right] + \boldsymbol{F} \tag{3.9}$$

where $\boldsymbol{u}_1 = [1 \ 0 \ \cdots \ 0]^T$ and $\boldsymbol{e}_k = \boldsymbol{d}_k - \boldsymbol{X}_k^T \boldsymbol{w}_{k-1}$; on the other hand, α_k, $\boldsymbol{Z}$, and $\boldsymbol{F}$ are defined in (3.6), (3.7), and (3.8), respectively. A regularization matrix $\delta \boldsymbol{I}$ with $0 < \delta \ll 1$ is added to $\boldsymbol{X}_k^T \boldsymbol{Z} \boldsymbol{X}_k$ in (3.5) and (3.9) in order to enable the computation of the inverse of $\boldsymbol{X}_k^T \boldsymbol{Z} \boldsymbol{X}_k$ for ill-conditioned problems.

The error-bound γ in (3.4) in the PCSMAP algorithm is obtained as [83]

$$\gamma = \begin{cases} \|\boldsymbol{e}_k\|_\infty - \nu \theta_k & \text{if } \|\boldsymbol{e}_k\|_\infty > \theta_k \\ \gamma_c & \text{otherwise} \end{cases} \tag{3.10}$$

where $0 < \nu \ll 1$, $\theta_k = Q\sigma_{e,k}$ with Q chosen such that $Q\sigma_{e,k} < \gamma_c$ in steady state.

The variance of the error signal is estimated as

$$\sigma_{e,k}^2 = \lambda \sigma_{e,k-1}^2 + (1 - \lambda) \text{median}(e_k^2, \ldots, e_{k-P+1}^2) \tag{3.11}$$

where $0 < \lambda < 1$ is the forgetting factor. If the initial value $\sigma_{e,0}^2$ in (3.11) is chosen to be large, we obtain $\|\boldsymbol{e}_k\|_\infty < \theta_k$ during transience and, therefore, the algorithm will work with the error bound $\gamma = \gamma_c$, which leads to a faster convergence. However, since $\sigma_{e,k}^2 \approx \sigma_v^2$ holds true in steady sate, we obtain $\|\boldsymbol{e}_k\|_\infty > \theta_k$ and as a result the algorithm would work with error bound $\|\boldsymbol{e}_k\|_\infty - \nu\theta_k$ which leads to a reduced steady-state misalignment. In addition, error bound $\|\boldsymbol{e}_k\|_\infty - \nu\theta_k$ suppresses the impulsive noise-corrupted error signal. Under sudden system disturbances, the estimate $\sigma_{e,k}^2$ in (3.11) tends to grow in which case the algorithm would again work with error bound γ_c, which leads to fast re-adaptation.

3.3 Simulation Results

In this section the performance of the PCSMAP algorithm is compared with that of the constrained NLMS (CNLMS) algorithm [32], the CAP [27], and the CSMAP algorithm [27] in several experiments as detailed below.

3.3.1 Linear-Phase Plant Identification Application

In this experiment, the unknown system was an FIR filter with length $M = 57$. The input signal was a zero-mean Gaussian noise signal with variance 10^{-4} and was colored by an IIR filter with transfer function [19]

$$H(z) = \frac{1}{1 - 0.95z^{-1} - 0.19z^{-2} - 0.09z^{-3} + 0.5z^{-4}} \tag{3.12}$$

The eigenvalue spread ratio of the colored input signal of 10000 samples was obtained as 2738 by using the ensemble average of the autocorrelation matrix given by (1.28) over 1000 independent trials with $\lambda = 1 - 2^{-15}$ and $\boldsymbol{R}_0 = 10^{-4}\boldsymbol{I}$. The constraint matrix was chosen as

$$\boldsymbol{C} = [\boldsymbol{I}\ \boldsymbol{0}\ -\boldsymbol{J}] \tag{3.13}$$

where $\boldsymbol{I}$ is the identity matrix of dimension $[(M-1)/2, (M-1)/2]$, vector $\boldsymbol{0}$ is of dimension $(M-1)/2$, matrix $\boldsymbol{J}$ is of dimension $[(M-1)/2, (M-1)/2]$ whose rows are obtained by flipping the rows of $\boldsymbol{I}$ from left to right. The constraint vector $\boldsymbol{f}$ of dimension $(M-1)/2$ is chosen as a zero vector. With this choice of $\boldsymbol{C}$ and $\boldsymbol{f}$, the weight-vector updates in each iteration would maintain a linear phase. The learning curves obtained in 1000 independent trials by using the CNLMS, CAP, CSMAP and the PCSMAP algorithms are illustrated in Fig. 3.1. As can be seen, the PCSMAP algorithm yields a significantly reduced steady-state misalignment as compared to the other algorithms. In order to investigate the robustness of the PCSMAP algorithm with respect to impulsive noise and re-adaptation capability, we repeated the same experiment except that all FIR filter coefficients were multiplied by -1 at iteration 1500. The impulsive noise added to the desired signal was modeled as $\phi_k = \omega_k \chi_k$

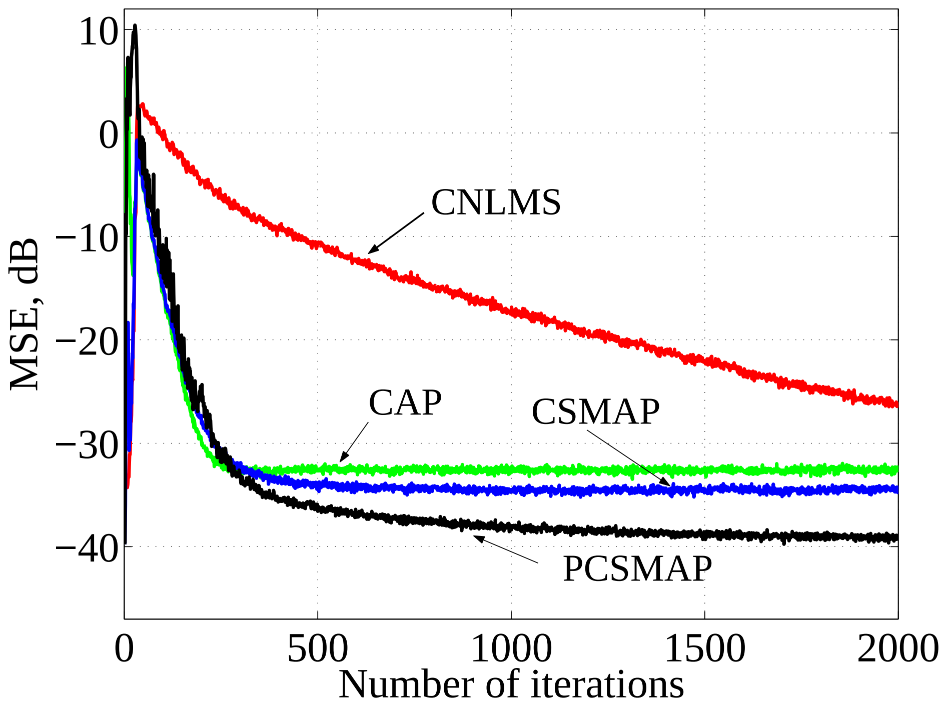

Figure 3.1: Learning curves in linear-phase plant identification application without impulsive noise using $L = 8$, $\sigma_{e,0}^2 = 100$, $\lambda = 0.95$, $Q = 1.88$ $\nu = 0.05$, $\gamma_c = \sqrt{5\sigma_v^2}$, $P = 15$.

where ω_k is a Bernoulli process with $P(\omega_k = 1) = 0.001$ and χ_k is a zero-mean Gaussian signal with variance $\sigma_\chi^2 = 1,000\sigma_y^2$ where σ_y^2 is the power of the uncorrupted output signal. The learning curves obtained in 1000 independent trials by using the CNLMS, CAP, CSMAP, and PCSMAP algorithms are illustrated in Fig. 3.2. As can be seen, the PCSMAP algorithm performs much better than the other algorithms. Constraining the phase would be useful in applications where error due to phase of the error signal becomes more prominent than the error due to the amplitude of the error signal.

3.3.2 Sinusoid Filtering with Minimum Variance Application

In this experiment, the weight-vector update is constrained to pass only the frequency components of 0.1 rad/sec and 0.25 rad/sec of the signal [99]

$$x(k) = a[\sin(0.3k\pi) + \sin(0.325k\pi) + \sin(0.7k\pi)] + n(k) \tag{3.14}$$

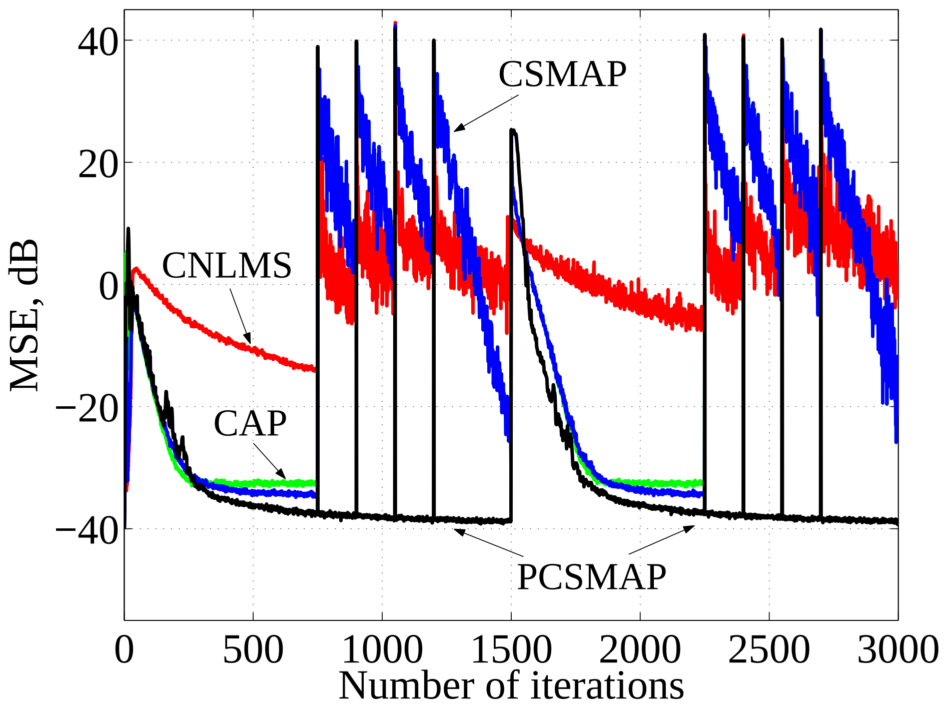

Figure 3.2: Learning curves in linear-phase plant identification application with impulsive noise using $L = 8$, $\sigma^2_{e,0} = 100$, $\lambda = 0.95$, $Q = 1.88$ $\nu = 0.05$, $\gamma_c = \sqrt{5\sigma_v^2}$, $P = 15$.

where $a = 1$, $n(k)$ is a white Gaussian noise signal with a variance 10^{-6} at the output of adaptive filter. The constraint matrix and constraint vector in this case become [99]

$$\boldsymbol{C} = \begin{bmatrix} 1 & \cos(0.2\pi) & \cdots & \cos[(M-1)0.2\pi] \\ 1 & \cos(0.5\pi) & \cdots & \cos[(M-1)0.5\pi] \\ 0 & \sin(0.2\pi) & \cdots & \sin[(M-1)0.2\pi] \\ 0 & \sin(0.5\pi) & \cdots & \sin[(M-1)0.5\pi] \end{bmatrix} \tag{3.15}$$

and

$$\boldsymbol{f}^T = [1\ 1\ 0\ 0] \tag{3.16}$$

respectively. The desired signal becomes zero. The amplitude a in (3.14) is switched to -1 at iteration 2000 to check the re-adaptation capability of the algorithms. The mean output error (MOE) curves obtained in 1000 independent trials are illustrated in Fig. 3.3. As can be seen, the PCSMAP algorithm yields a reduced steady-state misalignment as compared to the other algorithms.

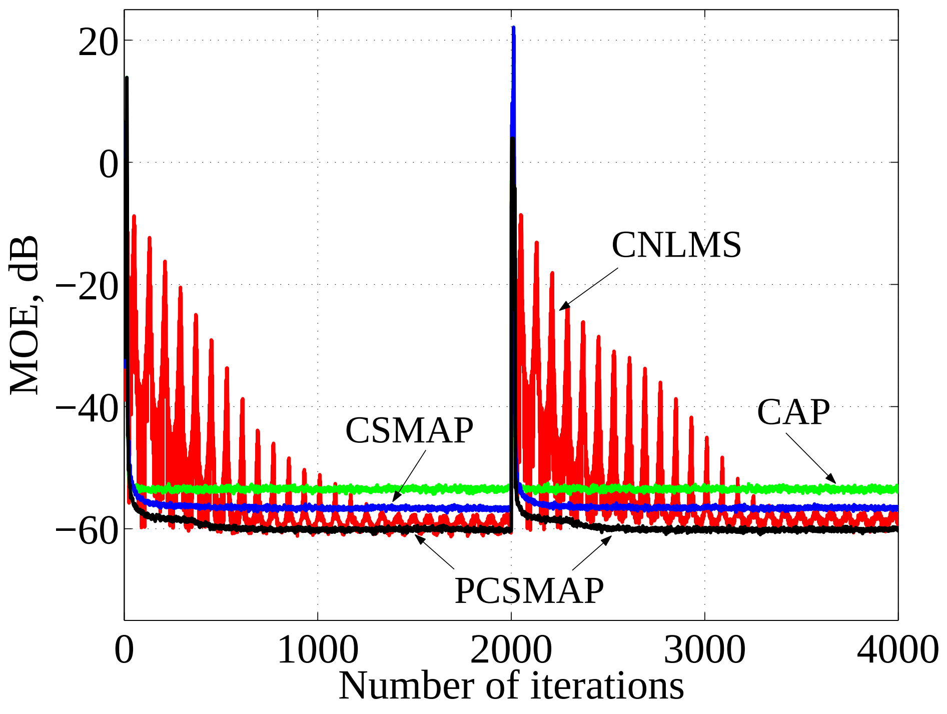

Figure 3.3: Learning curves in time-series filtering application with $L = 8$, $M = 14$, $\sigma_{e,0}^2 = 10$, $\lambda = 0.95$, $\nu = 0.05$, $Q = 1.88$, $\gamma_c = \sqrt{5\sigma_v^2}$, $P = 2$.

3.3.3 Interference Suppression in DS-CDMA Systems Application

The received signal in DS-CDMA systems with K users can be expressed in matrix form as [32]

$$\boldsymbol{x}_k = \boldsymbol{S}\boldsymbol{A}\boldsymbol{b}_k + \boldsymbol{v}_k \tag{3.17}$$

where $\boldsymbol{S} = [\boldsymbol{s}_1 \ \boldsymbol{s}_2 \ \cdots \ \boldsymbol{s}_K]$ is the spreading matrix of size $[M, K]$, $A = \text{diag}[A_1 \ A_2 \ \cdots \ A_K]$ is the amplitude of the user signals, the entries of vector $\boldsymbol{b}_k$ of size $[K, 1]$ are information bits $\{1, -1\}$, and $\boldsymbol{v}_k$ is a white Gaussian noise sequence with a variance 10^{-2}. The amplitude A_1 of the desired user was set to unity and the amplitudes of the interfering users were set to 3. The constraint matrix and constraint vector in this application become $\boldsymbol{C} = \boldsymbol{s}_1^T$ and $\boldsymbol{f} = 1$, respectively. The spreading code $\boldsymbol{s}_i$ for each user A_i was taken to be a random unit-norm vector. The MOE curves obtained in 1000 independent trials are illustrated in Fig. 3.4. As can be seen, the PCSMAP algorithm yields a reduced steady-state misalignment as compared to the other algorithms.

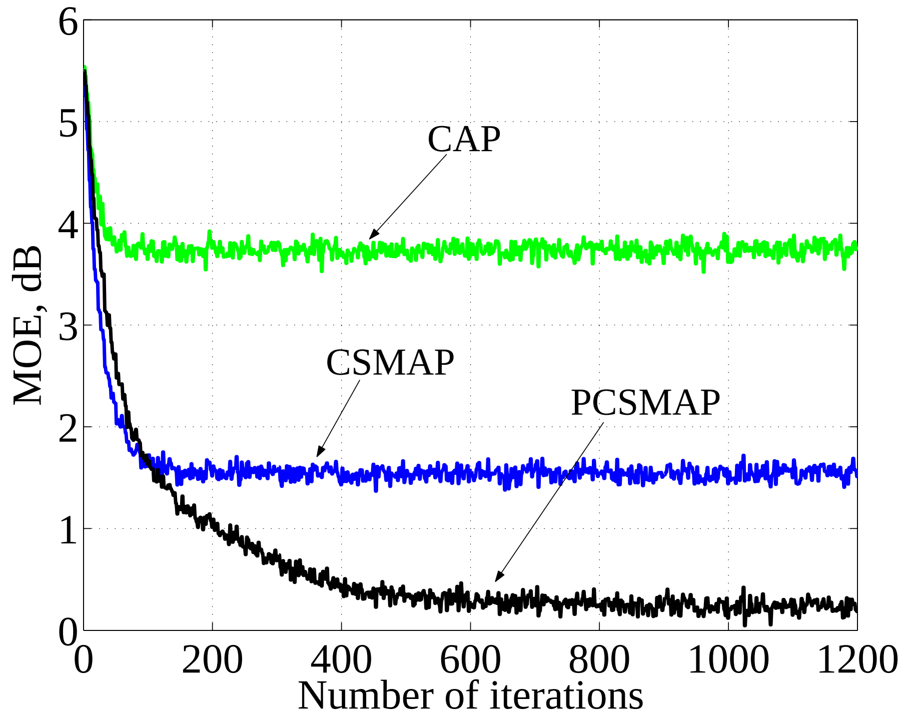

Figure 3.4: Learning curves in DS-CDMA interference suppression application. The parameters for the PCSMAP algorithm were: $L = 1$, $M = 32$, $K = 20$, $\lambda = 0.995$, $\sigma_{e,0}^2 = 10$, $\gamma_c = 1.1$, $Q = 0.925$, $\delta = 10^{-6}$, $\nu = 0.1$, $P = 15$. The CAP and CSMAP algorithms use the same L, P, M, δ. The CSMAP algorithm uses $\gamma = 1.1$.

3.4 Conclusions

A new robust set-membership CAP adaptation algorithm was developed. The proposed algorithm uses two error bounds. The switching of the two error bounds is obtained by using a variance estimator for the error signal. With these two error bounds, improved robustness with respect to impulsive noise, a significantly reduced steady-state misalignment, and slightly improved re-adaptation capability can be achieved by using the proposed CSMAP algorithm as compared to using the NCLMS, CAP, and SMCAP algorithms.

In the next three chapters, we discuss several improved Newton-type adaptation algorithms.

Chapter 4

Robust Recursive Least-Squares Adaptation Algorithm for Impulsive-Noise Environments

4.1 Introduction

Robustness in adaptive filters in impulsive-noise environments is achieved in a number of ways [2]. In [63, 64, 67], adaptive-filter robustness is considered as insensitivity to impulsive noise and in [68] it is deemed to be the capability of an adaptive filter to reconverge to the steady-state solution at the same rate of convergence as before the disturbance. The robust algorithms in [63, 64] use the Hampel three-part redescending M-estimate objective function and that in [67] uses the Huber two-part M-estimate objective function. In [63, 64, 67], the median absolute deviation (MAD) [66] is used to estimate the variance of the error signal in order to determine appropriate threshold values that are used to control the learning rate for an impulsive-noise-corrupted error signal. In [68], the instantaneous power of the scaled error signal is lowpass filtered and then used to switch the step size of the algorithm between two levels, one of which suppresses the error signal corrupted by impulsive noise during the adaptation of the weight vector.

In this chapter, we propose a new robust RLS (RRLS) adaptation algorithm that yields an optimal solution of the scaled least-squares objective function [84]. The proposed algorithm is robust with respect to impulsive noise as well as long bursts of impulsive noise in the sense that it converges back to the steady state much faster than during the initial convergence. The proposed algorithm also tracks sudden system disturbances. Simulation results show that the proposed algorithm achieves improved robustness and better re-adaptation capability as compared to the conventional RLS and recursive least-M estimate (RLM) algorithms reported in [63].

4.2 Proposed Robust RLS Algorithm

Two slightly different versions of the proposed RRLS algorithm are possible as detailed below, one for stationary and the other for nonstationary environments.

4.2.1 RRLS Algorithm for Stationary Environments

Scaled least-squares algorithms obtain the optimal weight vector $\boldsymbol{w}_k$ at iteration k by solving the optimization problem

$$\underset{\boldsymbol{w}_k}{\text{minimize}} \quad \textstyle\sum_{i=1}^{k} q_i(d_i - \boldsymbol{w}_k^T \boldsymbol{x}_i)^2 \tag{4.1}$$

where d_i is the desired signal, $\boldsymbol{x}_i$ is the input signal vector, and q_i is a nonnegative scalar at iteration i. Each of vectors $\boldsymbol{w}_k$ and $\boldsymbol{x}_i$ is of dimension M. The solution of (4.1) is achieved by solving the normal equations which are obtained by setting the gradient of the objective function in (4.1) with respect to $\boldsymbol{w}_k$ to zero. The input signal autocorrelation matrix, $\boldsymbol{R}_k$, and crosscorrelation vector, $\boldsymbol{p}_k$, at iteration k are given by

$$\boldsymbol{R}_k = \lambda_f \boldsymbol{R}_{k-1} + \delta_k \boldsymbol{x}_k \boldsymbol{x}_k^T \tag{4.2}$$

$$\boldsymbol{p}_k = \lambda_f \boldsymbol{p}_{k-1} + \delta_k \boldsymbol{x}_k d_k \tag{4.3}$$

where $\boldsymbol{R}_k$ and $\boldsymbol{p}_k$ are of dimensions $[M, M]$ and M, respectively, and $0 \ll \lambda_f < 1$. Parameter λ_f is a prespecified fixed forgetting factor and δ_k is a nonnegative scalar. The normal equations of the problem in (4.1) can be expressed in matrix form as

$$\boldsymbol{R}_k \boldsymbol{w}_k = \boldsymbol{p}_k \tag{4.4}$$

Using the matrix inversion lemma [3]-[2] in (4.2), we obtain the update equation of the inverse of the autocorrelation matrix as

$$\boldsymbol{S}_k = \frac{1}{\lambda_f}\left(\boldsymbol{S}_{k-1} - \frac{1}{\dfrac{\lambda_f}{\delta_k} + \boldsymbol{x}_k^T \boldsymbol{S}_{k-1}\boldsymbol{x}_k}\boldsymbol{S}_{k-1}\boldsymbol{x}_k\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\right) \tag{4.5}$$

Now using (4.5) in (4.4), the update equation of the weight vector is obtained as

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \frac{1}{\dfrac{\lambda_f}{\delta_k} + \boldsymbol{x}_k^T \boldsymbol{S}_{k-1}\boldsymbol{x}_k}\boldsymbol{S}_{k-1}\boldsymbol{x}_k e_k \tag{4.6}$$

where

$$e_k = d_k - \boldsymbol{w}_{k-1}^T\boldsymbol{x}_k \tag{4.7}$$

is the *a priori* error.

In impulsive-noise environments, the L_1 norm of the *gain vector*, i.e., $\boldsymbol{p}_k - \lambda\boldsymbol{p}_{k-1}$, given by

$$\|\boldsymbol{p}_k - \lambda\boldsymbol{p}_{k-1}\|_1 = \|\delta_k\boldsymbol{x}_k d_k\|_1 \tag{4.8}$$

undergoes a sudden increase when d_k or $\boldsymbol{x}_k$ are corrupted by impulsive noise. As a result, the L_1 norm of $\boldsymbol{p}_k$ is also increased which would, in turn, increase the L_1 norm of $\boldsymbol{w}_k$ in (4.4). The effect of impulsive noise on (4.3) can be suppressed by imposing a time-varying upper bound γ_k on the L_1 norm of the gain vector in (4.8). In other words, we choose δ_k such that the update of the crosscorrelation vector in (4.3) satisfies the condition

$$\|\boldsymbol{p}_k - \lambda\boldsymbol{p}_{k-1}\|_1 \le \gamma_k \tag{4.9}$$

Parameter γ_k is chosen as

$$\gamma_k = \left|\frac{\nu d_k}{e_k}\right| \tag{4.10}$$

where $0 < \nu \le 1$ for all k on the basis of extensive simulations. The condition in (4.9) is satisfied if δ_k is chosen as

$$\delta_k = \frac{\nu}{\|\boldsymbol{x}_k e_k\|_1} \tag{4.11}$$

As can be seen, δ_k can be greater than unity which would affect the convergence performance of the adaptive filter. To circumvent this problem, we use

$$\delta_k = \min\left(1, \frac{\nu}{\|\boldsymbol{x}_k e_k\|_1}\right) \tag{4.12}$$

With $\delta_k = 1$, the update equations in (4.5) and (4.6) become identical with those of the conventional RLS adaptation algorithm. The value of δ_k given by (4.12) will also bound the L_1 norm of the *gain matrix*, i.e., $\boldsymbol{R}_k - \lambda\boldsymbol{R}_{k-1}$, given by

$$\begin{aligned}\|\boldsymbol{R}_k - \lambda\boldsymbol{R}_{k-1}\|_1 &= \left\|\delta_k \boldsymbol{x}_k \boldsymbol{x}_k^T\right\|_1 \\ &= \min\left(\|\boldsymbol{x}_k\|_\infty \|\boldsymbol{x}_k\|_1, \frac{\nu}{|e_k|}\|\boldsymbol{x}_k\|_\infty\right)\end{aligned} \tag{4.13}$$

As can be seen, for an impulsive-noise corrupted e_k, the L_1 norm of the *gain matrix* would be significantly reduced. Since the probability that $\delta_k = 1$ during the transient state is high and the convergence of the RLS algorithm is fast, the initial convergence of the proposed RRLS algorithm would also be fast. In addition, the proposed RRLS algorithm would work with $\delta_k = 1$ during steady state as the amplitude of the error signal, e_k, becomes quite low during steady state. Consequently, the steady-state misalignment of the proposed RRLS algorithm would be similar to that of conventional RLS adaptation algorithms. However, when an impulsive noise-corrupted e_k occurs, we obtain $|d_k| \approx |e_k|$ and $\delta_k = \nu / \|\boldsymbol{x}_k e_k\|_1$ which would force the L_1 norm of the gain vector in (4.8) and the L_1 norm of the *gain matrix* in (4.13) to be bounded by $\gamma_k \approx \nu$ and $\|\boldsymbol{x}_k\|_\infty \nu/|e_k|$, respectively. As a result, the L_1 norm of the weight vector $\boldsymbol{w}_k$ in (4.4) would also remain bounded as discussed below.

The L_1 norm of the differential-weight vector of the conventional RLS algorithm given by

$$\triangle\boldsymbol{w}_k = \boldsymbol{w}_k - \boldsymbol{w}_{k-1} \tag{4.14}$$

is obtained as

$$\|\triangle\boldsymbol{w}_k\|_1 = \frac{|e_k|\,\|\boldsymbol{S}_{k-1}\boldsymbol{x}_k\|_1}{\|\lambda + \boldsymbol{x}_k\boldsymbol{S}_{k-1}\boldsymbol{x}_k^T\|_1} \tag{4.15}$$

by using (4.6) in (4.14) with $\delta_k = 1$. As can be seen, the L_1 norm of the differential-weight vector in the conventional RLS algorithm increases abruptly for an impulsive noise corrupted e_k. Similarly, the L_1 norm of the differential-weight vector in the proposed RRLS algorithm for the case of an impulsive noise corrupted error signal,

e_k, is obtained by using (4.11) and (4.6) in (4.14) as

$$\|\triangle \boldsymbol{w}_k\|_1 = \frac{\nu \|\boldsymbol{S}_{k-1}\boldsymbol{x}_k\|_1}{\|\boldsymbol{x}_k\|_1 \|\lambda + \delta_k \boldsymbol{x}_k \boldsymbol{S}_{k-1}\boldsymbol{x}_k^T\|_1} \tag{4.16}$$

As can be seen, the L_1 norm given by (4.16) would be much less than that in (4.15) since e_k cannot perturb $\boldsymbol{S}_{k-1}$. Although δ_k would become less than one in such a situation, its effect is significantly reduced by $\|\boldsymbol{x}_k\|_1$ in (4.16). In addition, $\|\boldsymbol{x}_k\|_1$ would significantly reduce the effect of an impulsive-noise corrupted $\boldsymbol{x}_k$ on $\|\triangle \boldsymbol{w}_k\|_1$ in (4.16) compared to that in (4.15). It should also be noted that the duration of $\boldsymbol{x}_k$ and e_k would have no effect on (4.16). In other words, the proposed RRLS algorithm would exhibit robust performance with respect to a long burst of impulsive noise. Using the well known vector-norm inequality

$$\frac{1}{\sqrt{M}} \|\triangle \boldsymbol{w}_k\|_1 \leq \|\triangle \boldsymbol{w}_k\|_2 \leq \|\triangle \boldsymbol{w}_k\|_1 \tag{4.17}$$

and (4.16), we note that the L_2 norm of the differential-weight vector would also remain bounded and hence the L_2 norm of $\boldsymbol{w}_k$ in the proposed RRLS algorithm would also be robust with respect to the amplitude and duration of the impulsive-noise corrupted e_k.

4.2.2 RRLS Algorithm for Nonstationary Environments

The above RRLS algorithm, like other RLS algorithms, cannot track sudden system disturbances as λ_f is chosen to be very close to unity in order to achieve a reduced steady-state misalignment. To overcome this problem, we use time-varying parameters λ_k and δ_k defined as

$$\lambda_k = \max\left[0.1, \min\left(\lambda_f, \frac{\theta_{1,k}\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k}{\theta_{2,k} - \theta_{1,k} + \theta_{1,k}\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k}\right)\right] \tag{4.18}$$

$$\delta_k = 1 - \lambda_k \tag{4.19}$$

In (4.18), $\theta_{1,k}$ should be greater than $\theta_{2,k}$ in order to render the proposed RRLS algorithm applicable to nonstationary environments. Suitable values for $\theta_{1,k}$ and $\theta_{2,k}$ that were found to give good results in practice are $\theta_{1,k} = 2.24\sigma_{1,e}$ and $\theta_{2,k} = \sigma_{2,e}$. Constant 2.24 is an empirical constant which is chosen to ensure that the probability that $\lambda_k \neq \lambda_f$ is of the order of 0.001. This would ensure that under sudden system disturbances, λ_k would be reduced momentarily and then be quickly returned to the value λ_f in order to maintain the re-adaptation capability of the algorithm. The variances of the error signal, $\sigma^2_{1,e}$ and $\sigma^2_{2,e}$, in iteration k are estimated as

$$\sigma^2_{1,k} = \beta\sigma^2_{1,k-1} + (1-\beta)\min\left[\sigma^2_{1,k-1}, \mathrm{median}(\boldsymbol{g}_k)\right] \tag{4.20}$$

$$\sigma^2_{2,k} = \varsigma\sigma^2_{2,k-1} + (1-\varsigma)\mathrm{median}(\boldsymbol{g}_k) \tag{4.21}$$

where $\boldsymbol{g}_k^T = \left[e_k^2 + \epsilon, \ldots, e^2_{k-P+1} + \epsilon\right]$ is a vector of dimension P, $\epsilon \approx 0$ is a very small positive scalar, and $0 < \beta < 1$ and $0 < \varsigma < 1$. Parameters β and ς are referred to as *memory factors* in the literature. In the proposed algorithm, we use λ_k and δ_k given in (4.18) and (4.19), respectively, only when $\sqrt{\min(\boldsymbol{g}_k)} > 4\sigma_{1,k}$. Otherwise, we use λ_f and δ_k as given in (4.12).

If $\sigma^2_{1,0}$ is chosen to be very large, then we would get $\sqrt{\min(\boldsymbol{g}_k)} < 4\sigma_{1,k}$ during the transient state and, therefore, the algorithm would work with λ_f and δ_k as given in (4.12). As a result, the transient state would die out quickly in which case $\sigma_k^2 \approx \sigma_v^2$ at steady state. On the other hand, for sudden system disturbances, we would get $\sqrt{\min(\boldsymbol{g}_k)} \gg 4\sigma_{1,k}$ in which case the algorithm would work with λ_k and δ_k given in (4.18) and (4.19), respectively. In such a situation, λ_k momentarily becomes significantly less than λ_f as $\theta_{2,k} \gg \theta_{1,k}$ and shortly afterwards λ_k becomes equal to λ_f as $\theta_{2,k}$ becomes less than $\theta_{1,k}$. As a result, improved re-adaptation capability is achieved in nonstationary environments.

4.2.3 Discussion

The two versions of the proposed algorithm essentially solve the minimization problem

$$\underset{\boldsymbol{w}_k}{\text{minimize}}\ J_{\boldsymbol{w}_k} = \textstyle\sum_{i=1}^k \delta_i \prod_{j=i+1}^k \lambda_j (d_i - \boldsymbol{w}_k^T\boldsymbol{x}_i)^2 + 0.5\left(\prod_{i=1}^k \lambda_i\right)\boldsymbol{w}_k^T\boldsymbol{R}_0\boldsymbol{w}_k \tag{4.22}$$

and they can be implemented in terms of the algorithm summarized in Table 4.1 [84].

For stationary environments, the proposed algorithm entails $3M^2+4M+5$ multiplications and $2M^2+2M+2$ additions per iteration where M is the dimension of the weight vector. On the other hand, for nonstationary environments, $3M^2+4M+13$ multiplications and $2M^2+2M+5$ additions per iteration are required. The conven-

Table 4.1: Proposed RRLS Algorithm

```
Given $d_k$ and $\boldsymbol{x}_k$ choose $P$, $\lambda_f$, $\boldsymbol{S}_0 = \epsilon^{-1}\boldsymbol{I}$, and compute
  $e_k = d_k - \boldsymbol{w}_{k-1}^T\boldsymbol{x}_k$
  $\boldsymbol{t}_k = \boldsymbol{S}_{k-1}\boldsymbol{x}_k$
  $\tau_k = \boldsymbol{x}_k^T\boldsymbol{t}_k$
  $\tilde{\tau}_k = \frac{\lambda_k}{\delta_k} + \tau_k$
  $\tilde{\boldsymbol{t}}_k = \frac{1}{\tilde{\tau}_k}\boldsymbol{t}_k$
  $\boldsymbol{S}_k = \frac{1}{\lambda_k}\left(\boldsymbol{S}_{k-1} - \tilde{\boldsymbol{t}}_k\boldsymbol{t}_k^T\right)$
  $\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + e_k\tilde{\boldsymbol{t}}_k$
For applications in stationary environments, compute
  $\lambda_k = \lambda_f$
  $\|\boldsymbol{x}_k\|_1 = \|\boldsymbol{x}_{k-1}\|_1 + |x_k| - |x_{k-M}|$
  $\delta_k = \min\left(1, \frac{\nu}{\|\boldsymbol{x}_k\|_1|e_k|}\right)$
For applications in nonstationary environments, compute
  $\boldsymbol{g}_k = \left[e_k^2 + \epsilon, \quad \boldsymbol{g}_{k-1}^T(1, 1:P-1)\right]^T$
  $\mathcal{C} = \text{median}(\boldsymbol{g}_k)$
  $\sigma_{1,k}^2 = \beta\sigma_{1,k-1}^2 + (1-\beta)\min(\sigma_{1,k-1}^2, \mathcal{C})$
  $\sigma_{2,k}^2 = \varsigma\sigma_{2,k-1}^2 + (1-\varsigma)\mathcal{C}$
  if $\sqrt{\min(\boldsymbol{g}_k)} > 4\sigma_{1,k}$ let
    $\lambda_k = \max\left[0.1, \min\left(\lambda_f, \frac{\theta_{1,k}\tau_k}{\theta_{2,k} - \theta_{1,k} + \theta_{1,k}\tau_k}\right)\right]$
    $\delta_k = 1 - \lambda_k$
  else let
    $\lambda_k = \lambda_f$
    $\delta_k = \min\left(1, \frac{\nu}{\|\boldsymbol{x}_k\|_1|e_k|}\right)$
  end
```

tional RLS algorithm requires $3M^2+4M+2$ multiplications and $2M^2+2M$ additions whereas the RLM algorithm requires $3M^2+4M+11$ multiplications and $2M^2+2M+2$ additions. Evidently, for values of M greater than 5, the computational complexity of the proposed RRLS algorithm is similar to that of the RLS and RLM algorithms.

4.3 Simulation Results

In this section, the proposed RRLS (PRRLS) algorithm is compared with the conventional RLS algorithm and the RLM algorithm [63] in terms of robustness and its capability in tracking abrupt disturbances.

The first experiment deals with a system identification application in a stationary environment. The weight vector of the unknown system, $\boldsymbol{w}_{opt}$, was obtained using MATLAB commands $\boldsymbol{h} = \text{fir1}(M-1, \omega_n)$ and $\boldsymbol{w}_{opt} = \boldsymbol{h}/\text{norm}(\boldsymbol{h}, 2)$ with $M = 37$ and $\omega_n = 0.3$. The input signal was a zero-mean white Gaussian noise signal with unity variance and it was colored by an IIR filter with a single pole at 0.95. The eigenvalue spread ratio of the colored input signal of 10000 samples was obtained as 837 by using the ensemble average of the autocorrelation matrix given by (1.28) over 1000 independent trials with $\lambda = 1 - 2^{-15}$ and $\boldsymbol{R}_0 = 10^{-4}\boldsymbol{I}$. The measurement noise added to the desired signal was a zero-mean white Gaussian noise signal with variances $\sigma_v^2 = 10^{-3}$ and 10^{-6} to achieve signal-to-noise ratios (SNRs) of 30 and 60 dB, respectively. The impulsive noise was generated as $\phi_k = \omega_k \chi_k$ where ω_k is a Bernoulli process with the probability that $\omega_k = 1$ is equal to $p = 0.001$, i.e., $P(\omega_k = 1) = 0.001$, and χ_k is a zero-mean Gaussian signal with variance $\sigma_\chi^2 = 10,000\sigma_y^2$ where σ_y^2 is the power of the uncorrupted output signal [68]. Parameter ν was set to unity for the PRRLS algorithm. The learning curves obtained in 1000 independent trials by using the conventional RLS and RLM algorithms and the PRRLS algorithm are illustrated in Figs. 4.1–4.2. As can be seen, the RLS and RLM algorithms are not robust with respect to a long burst of impulsive noise whereas the PRRLS algorithm is not affected by the impulsive noise.

In the second experiment, we repeated the first experiment with an identical experimental setup except that M was set to 17 and the impulsive noise was added to the adaptive-filter input signal rather than the desired signal. The eigenvalue spread ratio of the colored input signal of 10000 samples was obtained as 498 by using the ensemble average of the autocorrelation matrix given by (1.28) over 1000 independent trials with $\lambda = 1-2^{-15}$ and $\boldsymbol{R}_0 = 10^{-4}\boldsymbol{I}$. Due to the reduction in M, the

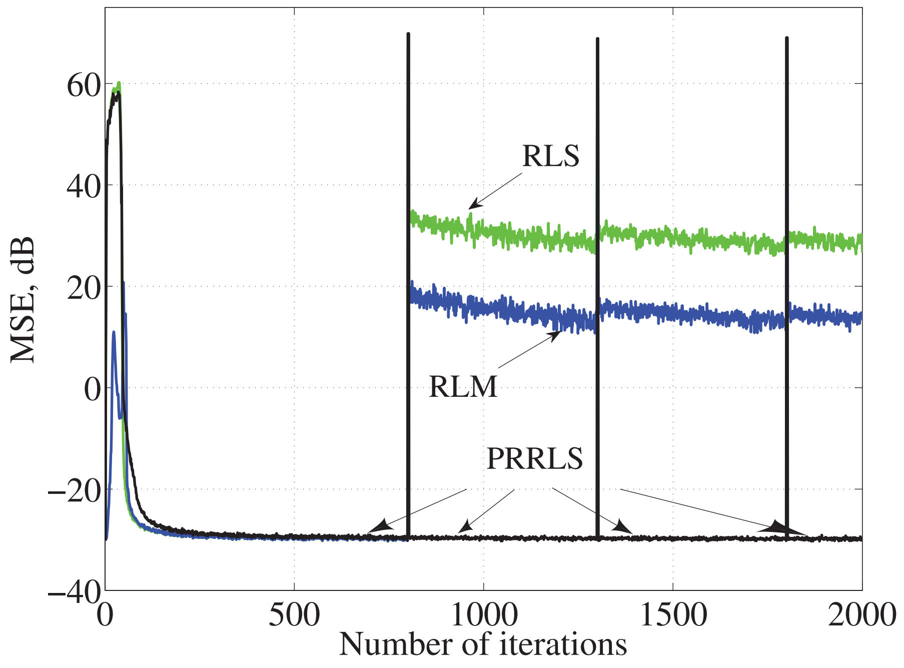

Figure 4.1: Learning curves with SNR = 30 dB, $\lambda_f = 0.999999$, $\boldsymbol{S}_0 = \epsilon^{-1}\boldsymbol{I}$ with $\epsilon = 10^{-12}$, $p = 0.001$, and $\boldsymbol{w}_0 = \boldsymbol{0}$ in all algorithms. The parameters for the RLM algorithm were $L = 5$, $\lambda_\sigma = 0.95$, $\xi = 1.960\sigma_e$, $\triangle_1 = 2.240\sigma_e$, $\triangle_2 = 2.576\sigma_e$ as suggested in [63]. Impulsive noise of duration $3T_S$ was added to the desired signal at iterations 800, 1300, 1800 where T_s is the sampling period.

eigenvalue spread ratio is reduced in this experiment. Parameter ν was set to 0.1 for the PRRLS algorithm. The learning curves obtained are illustrated in Figs. 4.3–4.4.

As can be seen, the PRRLS algorithm performs much better than the other algorithms in terms of steady-state misalignment and robustness with respect to impulsive noise.

The third experiment dealt with a system identification application in a nonstationary environment. The initial algorithm parameters were the same as those in the first experiment except that the order of the unknown system was increased to 63 and the variances of the measurement noise were changed to 10^{-4} and 10^{-7} to achieve SNRs of 40 and 70 dB, respectively. The eigenvalue spread ratio of the colored input signal of 10000 samples was obtained by using the ensemble average of the autocorrelation matrix given by (1.28) over 1000 independent trials with $\lambda = 1 - 2^{-15}$ and $\boldsymbol{R}_0 = 10^{-4}\boldsymbol{I}$ as 1081. The impulse response of the FIR filter was suddenly multiplied by -1 at iteration 1500. The re-adaptation capability of the robust algorithms in

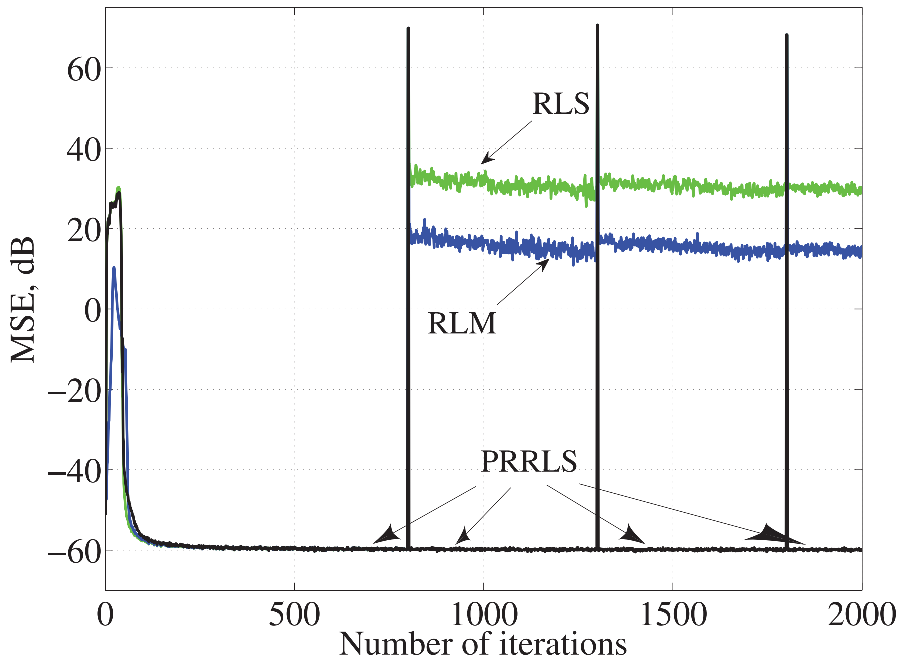

Figure 4.2: Learning curves with SNR = 60 dB, $\lambda_f = 0.999999$, $\boldsymbol{S}_0 = \epsilon^{-1}\boldsymbol{I}$ with $\epsilon = 10^{-12}$, $p = 0.001$, and $\boldsymbol{w}_0 = \boldsymbol{0}$ in all algorithms. The parameters for the RLM algorithm were $L = 5$, $\lambda_\sigma = 0.95$, $\xi = 1.960\sigma_e$, $\triangle_1 = 2.240\sigma_e$, $\triangle_2 = 2.576\sigma_e$ as suggested in [63]. Impulsive noise of duration $3T_S$ was added to the desired signal at iterations 800, 1300, 1800 where T_s is the sampling period.

[63, 64, 68, 67] was examined using a similar setting. Parameter ν was set to unity for the PRRLS algorithm. The learning curves obtained in 1000 independent trials by using the conventional RLS and RLM algorithms and the PRRLS algorithm are illustrated in Figs. 4.5–4.6.

As can be seen, the RLS and RLM algorithms cannot track sudden system changes whereas the PRRLS algorithm handles sudden system changes successfully and at the same time maintains its robustness with respect to impulsive noise.

4.4 Conclusions

A new robust RLS adaptation algorithm that performs well in impulsive noise environments has been proposed. The new algorithm uses the L_1 norm of the gain factor of the crosscorrelation vector to achieve robust performance against impulsive noise. In addition, the proposed algorithm uses a modified variance estimator to compute a

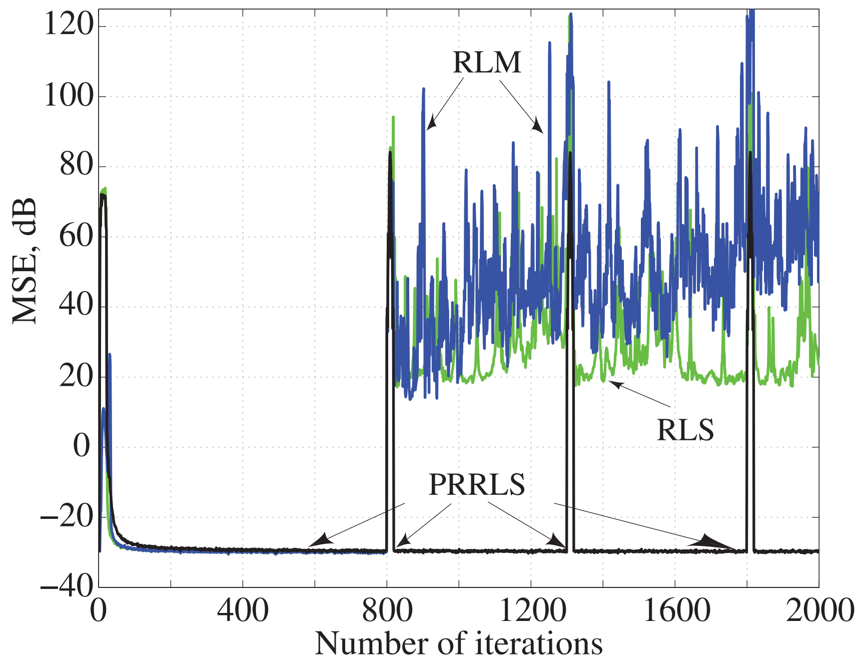

Figure 4.3: Learning curves with SNR = 30 dB, $\lambda_f = 0.999999$, $\boldsymbol{S}_0 = \epsilon^{-1}\boldsymbol{I}$ with $\epsilon = 10^{-12}$, $p = 0.001$, and $\boldsymbol{w}_0 = \boldsymbol{0}$ in all algorithms. The parameters for the RLM algorithm were $L = 5$, $\lambda_\sigma = 0.95$, $\xi = 1.960\sigma_e$, $\triangle_1 = 2.240\sigma_e$, $\triangle_2 = 2.576\sigma_e$ as suggested in [63]. Impulsive noise of duration $3T_S$ was added to the input signal at iterations 800, 1300, 1800 where T_s is the sampling period.

threshold that is used to obtain a variable forgetting factor λ_k which offers improved re-adaptation capability. Simulation results show that the proposed algorithm is robust against impulsive noise and offers better re-adaptation capability compared to the conventional RLS and RLM algorithms.

In the next chapter, we describe an improved quasi-Newton adaptation algorithm.

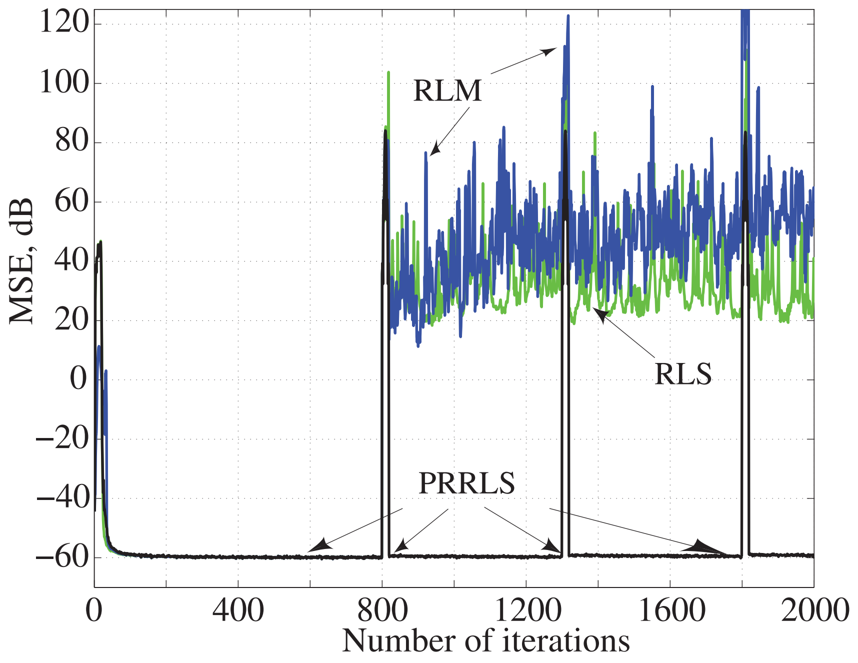

Figure 4.4: Learning curves with SNR = 60 dB, $\lambda_f = 0.999999$, $\boldsymbol{S}_0 = \epsilon^{-1}\boldsymbol{I}$ with $\epsilon = 10^{-12}$, $p = 0.001$, and $\boldsymbol{w}_0 = \boldsymbol{0}$ in all algorithms. The parameters for the RLM algorithm were $L = 5$, $\lambda_\sigma = 0.95$, $\xi = 1.960\sigma_e$, $\triangle_1 = 2.240\sigma_e$, $\triangle_2 = 2.576\sigma_e$ as suggested in [63]. Impulsive noise of duration $3T_S$ was added to the input signal at iterations 800, 1300, 1800 where T_s is the sampling period.

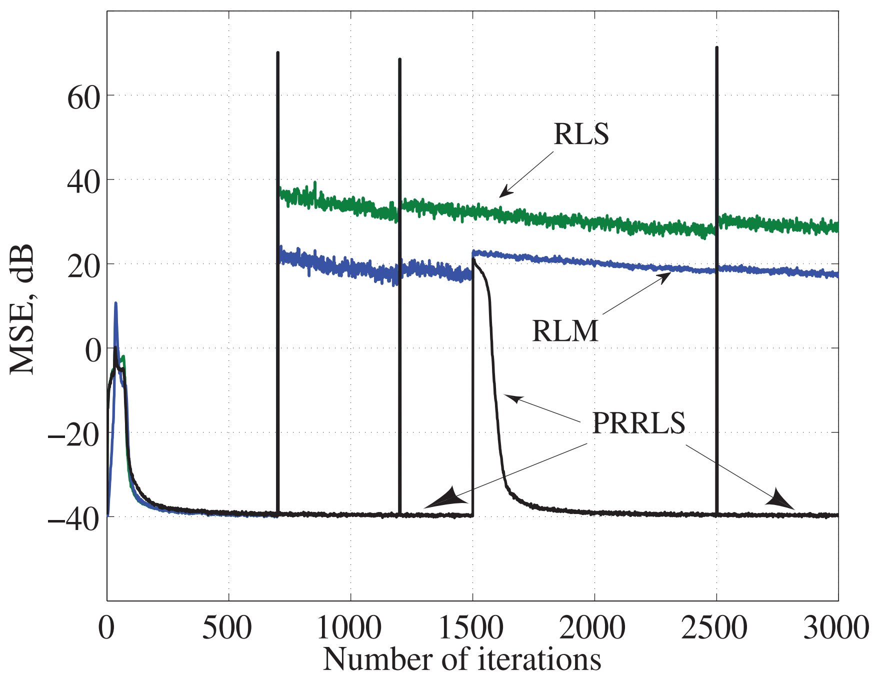

Figure 4.5: Learning curves with SNR = 40 dB, $P = 5$, $\sigma_{1,0} = \sigma_{2,0} = 1000$, $\beta = 0.99$, $\varsigma = 0.95$, for the PRRLS algorithm. Impulsive noise of duration $3T_S$ was added to the desired signal at iterations 700, 1200, 2500. The initial algorithm parameters were set to $\lambda_f = 0.999999$, $\boldsymbol{S}_0 = \epsilon^{-1}\boldsymbol{I}$ with $\epsilon = 10^{-4}$, and $\boldsymbol{w}_0 = \boldsymbol{0}$ in all algorithms.

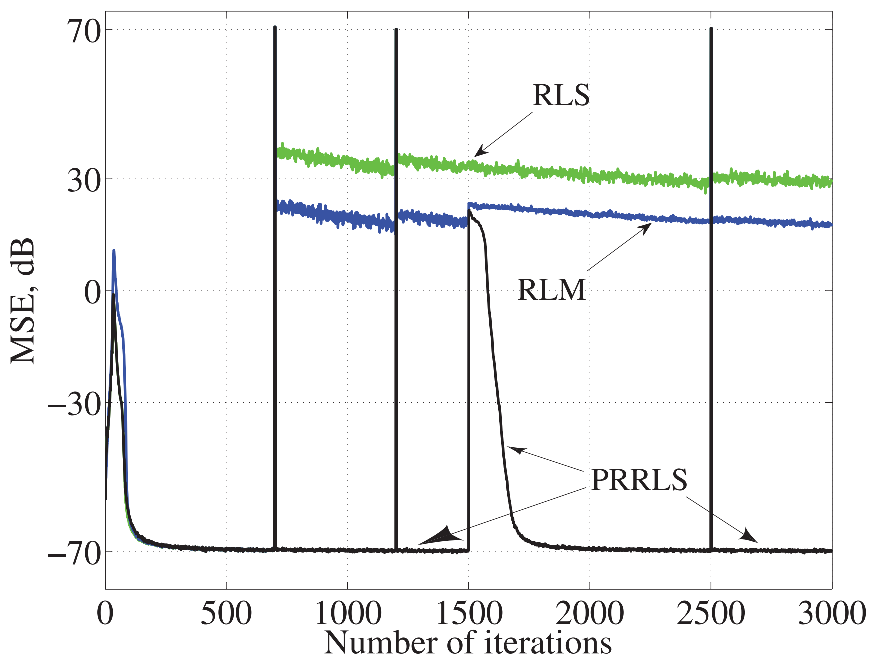

Figure 4.6: Learning curves with SNR = 70 dB, $P = 5$, $\sigma_{1,0} = \sigma_{2,0} = 1000$, $\beta = 0.99$, $\varsigma = 0.95$, for the PRRLS algorithm. Impulsive noise of duration $3T_S$ was added to the desired signal at iterations 700, 1200, 2500. The initial algorithm parameters were set to $\lambda_f = 0.999999$, $\boldsymbol{S}_0 = \epsilon^{-1}\boldsymbol{I}$ with $\epsilon = 10^{-4}$, and $\boldsymbol{w}_0 = \boldsymbol{0}$ in all algorithms.

Chapter 5

Improved Quasi-Newton Adaptation Algorithm

5.1 Introduction

The conventional recursive least-squares (CRLS) algorithm converges much faster than algorithms of the steepest-descent family [3]. However, it can become unstable and offers poor performance in tracking abrupt system disturbances especially if a large forgetting factor is used. The known quasi-Newton (KQN) algorithm reported in [57, 58] offers better numerical robustness whereas the LMS-Newton (LMSN) algorithms reported in [59] offer better convergence performance than the CRLS algorithm. The numerical robustness of the quasi-Newton (QN) algorithm reported in [57, 58] is achieved by using a biased estimate of the autocorrelation matrix, which can reduce the tracking capability of the algorithm relative to that of the RLS algorithm [64].

In this chapter, we propose an improved version of the QN algorithm reported in [57, 58] that incorporates data-selective adaptation [85]. The proposed QN (PQN) algorithm takes fewer weight updates to converge and yields a reduced steady-state misalignment relative to the KQN algorithm in [57, 58]. These features of the new algorithm are demonstrated through MATLAB simulations in stationary and non-stationary environments. Simulations also show that the PQN algorithm, like the KQN algorithm, is quite robust with respect to roundoff errors in fixed-precision implementations.

5.2 Proposed Quasi-Newton Algorithm

The objective of the proposed adaptation algorithm is to generate a series of weight vectors that would eventually solve the optimization problem

$$\underset{\boldsymbol{w}}{\text{minimize}}\, E[(d_k - \boldsymbol{w}^T \boldsymbol{x}_k)^2] \tag{5.1}$$

recursively where $E[\cdot]$ is the expected value of $[\cdot]$, $\boldsymbol{x}_k$ is a vector of dimension M representing the input signal, d_k is the desired signal, and $\boldsymbol{w}$ is the weight vector which is also of dimension M. An approximate solution of the problem in (5.1) can be obtained by using the weight-vector update equation

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + 2\mu_k \boldsymbol{S}_{k-1} \boldsymbol{x}_k e_k \tag{5.2}$$

where μ_k is the step size, $\boldsymbol{S}_{k-1}$ is a positive definite matrix of size $M \times M$, and

$$e_k = d_k - y_k \tag{5.3}$$

is the *a priori* error for the output signal $y_k = \boldsymbol{w}_{k-1}^T \boldsymbol{x}_k$. If $\boldsymbol{S}_{k-1}$ in (5.2) is chosen as the $M \times M$ identity matrix, then the update equation in (5.2) would minimize the objective function

$$J_{\boldsymbol{w}_{k-1}} = (d_k - \boldsymbol{w}_{k-1}^T \boldsymbol{x}_k)^2 \tag{5.4}$$

with respect to the steepest-descent direction and a series of updates would eventually yield an approximate solution of the problem in (5.1). Other choices of $\boldsymbol{S}_{k-1}$ would entail different search directions but would serve the same purpose as long as $\boldsymbol{S}_{k-1}$ remains positive definite. In order to use an approximation to the Newton direction $\boldsymbol{S}_{k-1}$ in (5.2), we obtain $\boldsymbol{S}_{k-1}$ by using the gradient of $J_{\boldsymbol{w}_{k-1}}$ in (5.4) in the rank-one update formula of the classical QN optimization algorithm [70] given by

$$\boldsymbol{S}_k = \boldsymbol{S}_{k-1} - \frac{(\boldsymbol{\delta}_k - \boldsymbol{S}_{k-1}\boldsymbol{\rho}_k)(\boldsymbol{\delta}_k - \boldsymbol{S}_{k-1}\boldsymbol{\rho}_k)^T}{(\boldsymbol{\delta}_k - \boldsymbol{S}_{k-1}\boldsymbol{\rho}_k)^T \boldsymbol{\rho}_k} \tag{5.5}$$

where

$$\boldsymbol{\delta}_k = \boldsymbol{w}_k - \boldsymbol{w}_{k-1} \tag{5.6}$$

and

$$\boldsymbol{\rho}_k = \frac{\partial e_{k+1}^2}{\partial \boldsymbol{w}_k} - \frac{\partial e_k^2}{\partial \boldsymbol{w}_{k-1}} \tag{5.7}$$

This formula satisfies the Fletcher quasi-Newton condition $\boldsymbol{S}_k\boldsymbol{\rho}_k = \boldsymbol{\delta}_k$ [71]. From (5.3), we note that e_{k+1} requires future data $\boldsymbol{x}_{k+1}$ and d_{k+1}. To circumvent this problem, we use the *a posteriori* error

$$\epsilon_k = d_k - \boldsymbol{x}_k^T\boldsymbol{w}_k \tag{5.8}$$

in place of e_{k+1} in (5.7). As a first step in the proposed algorithm, we obtain a value of the step size μ_k in (5.2) by solving the optimization problem

$$\underset{\mu_k \in \mathcal{R}_+}{\text{minimize}} \;\; \left(|d_k - \boldsymbol{x}_k^T\boldsymbol{w}_k| - \gamma\right) \tag{5.9}$$

where γ is a prespecified error bound. The solution of this problem can be obtained as

$$\mu_k = \alpha_k \frac{1}{2\tau_k} \tag{5.10}$$

where $\tau_k = \boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k$ and

$$\alpha_k = \begin{cases} 1 - \dfrac{\gamma}{|e_k|} & \text{if } |e_k| > \gamma \\ 0 & \text{otherwise} \end{cases} \tag{5.11}$$

In effect, the step size μ_k is chosen to force the equality $|\epsilon_k| = \gamma$ whenever $|e_k| > \gamma$. Since μ_k in (5.10) forces ϵ_k to assume the value of the prespecified error bound, i.e.,

$$\epsilon_k = \gamma \ \text{sign}(e_k) \tag{5.12}$$

we obtain $\boldsymbol{\nabla}\epsilon_k^2 = \boldsymbol{0}$ and hence from (5.7), we have

$$\boldsymbol{\rho}_k = 2e_k\boldsymbol{x}_k \tag{5.13}$$

Vector $\boldsymbol{\delta}_k$, which is linearly dependent on $\boldsymbol{S}_{k-1}\boldsymbol{x}_k$, can be obtained by using (5.2) and (5.6). Since the equality in (5.12) is satisfied for each update, we can use the *a posteriori* error to obtain $\boldsymbol{\delta}_k$ as

$$\boldsymbol{\delta}_k = 2\gamma \cdot \text{sign}(e_k)\boldsymbol{S}_{k-1}\boldsymbol{x}_k \tag{5.14}$$

instead of the *a priori* error used in the known QN algorithm reported in [57, 58]. Now substituting $\boldsymbol{\rho}_k$ and $\boldsymbol{\delta}_k$ given by (5.13) and (5.14) in (5.5), we obtain an update

Table 5.1: PQN Algorithm

```
Initialize $\boldsymbol{w}_0 = \boldsymbol{0}$ and $\boldsymbol{S}_0 = \boldsymbol{I}$.
Choose $\gamma$.
Input $d_k$, $\boldsymbol{x}_k$ and compute
  $e_k = d_k - \boldsymbol{w}_{k-1}^T \boldsymbol{x}_k$
  if $|e_k| > \gamma$
      $\alpha_k = 1 - \frac{\gamma}{|e_k|}$
      $\boldsymbol{t}_k = \boldsymbol{S}_{k-1}\boldsymbol{x}_k$
      $\tau_k = \boldsymbol{x}_k^T \boldsymbol{t}_k$
      $\boldsymbol{g}_k = \alpha_k \boldsymbol{t}_k$
      $\boldsymbol{S}_k = \boldsymbol{S}_{k-1} - \frac{\boldsymbol{g}_k \boldsymbol{t}_k^T}{\tau_k}$
      $\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \frac{1}{\tau_k} e_k \boldsymbol{g}_k$
  end
```

equation for matrix $\boldsymbol{S}_k$ for the proposed QN algorithm as

$$\boldsymbol{S}_k = \boldsymbol{S}_{k-1} - \alpha_k \frac{\boldsymbol{S}_{k-1}\boldsymbol{x}_k\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}}{\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k} \tag{5.15}$$

Substituting μ_k given by (5.10) in (5.2), we obtain the corresponding weight-vector update equation as

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \alpha_k \frac{\boldsymbol{S}_{k-1}\boldsymbol{x}_k}{\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k} e_k \tag{5.16}$$

Updates are applied to $\boldsymbol{w}_{k-1}$ and $\boldsymbol{S}_{k-1}$ only if the *a priori* error exceeds the prespecified error bound γ as is done in the basic QN optimization algorithm [70]. Otherwise, no updates are applied. The PQN algorithm [85] can be implemented as detailed in Table 5.1.

The crosscorrelation vector of the PQN algorithm can be defined as

$$\boldsymbol{p}_k = \boldsymbol{p}_{k-1} + \frac{\alpha_k}{\tau_{p,k}} d_k \boldsymbol{x}_k \tag{5.17}$$

where $\tau_{p,k} = (1 - \alpha_k)\tau_k$. If we apply the matrix inversion lemma given in [2, 3] to (5.15), we obtain the input-signal autocorrelation matrix as

$$\boldsymbol{R}_k = \boldsymbol{R}_{k-1} + \frac{\alpha_k}{\tau_{p,k}} \boldsymbol{x}_k \boldsymbol{x}_k^T \tag{5.18}$$

Using (5.3) in (5.16), straightforward manipulation yields

$$\begin{aligned}\boldsymbol{w}_k = \boldsymbol{w}_{k-1} &+ \frac{\alpha_k}{\tau_{p,k}} d_k \boldsymbol{S}_{k-1}\boldsymbol{x}_k - \frac{\alpha_k}{\tau_k}\ \boldsymbol{S}_{k-1}\boldsymbol{x}_k\boldsymbol{x}_k^T\boldsymbol{w}_{k-1} \\ &- \frac{\alpha_k^2}{\tau_{p,k}} d_k \boldsymbol{S}_{k-1}\boldsymbol{x}_k\end{aligned} \tag{5.19}$$

Using the normal equation

$$\boldsymbol{R}_{k-1}\boldsymbol{w}_{k-1} = \boldsymbol{p}_{k-1}$$

in (5.19) and simplifying the expression obtained, we have

$$\boldsymbol{w}_k = \left(\boldsymbol{S}_{k-1} - \frac{\alpha_k}{\tau_k}\ \boldsymbol{S}_{k-1}\boldsymbol{x}_k\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\right)\left(\boldsymbol{p}_{k-1} + \frac{\alpha_k}{\tau_{p,k}} d_k \boldsymbol{x}_k\right) \tag{5.20}$$

Now using (5.15) and (5.17) in (5.20), we obtain

$$\boldsymbol{R}_k\boldsymbol{w}_k = \boldsymbol{p}_k \tag{5.21}$$

which comprises the normal equations of the objective function

$$J_{\boldsymbol{w}_k} = \sum_{i=1}^{k} \frac{\alpha_i}{\tau_{p,i}}\ (d_i - \boldsymbol{w}_k^T\boldsymbol{x}_i)^2 + \frac{1}{2}\boldsymbol{w}_k^T\boldsymbol{R}_0\boldsymbol{w}_k \tag{5.22}$$

where $\tau_{p,i} = (1 - \alpha_i)\boldsymbol{x}_i^T\boldsymbol{S}_{i-1}\boldsymbol{x}_i$. Hence the weight-vector update equation of the PQN algorithm in (5.16) minimizes the objective function in (5.22).

As can be seen, the objective function of the PQN algorithm is similar to that of the weighted LS or LMSN algorithm [59]. The updating formulas for the KQN algorithm are

$$\boldsymbol{S}_k = \boldsymbol{S}_{k-1} + \frac{[\mu_k - 1]}{\tau_k}\,\boldsymbol{S}_{k-1}\boldsymbol{x}_k\boldsymbol{x}_k^T\boldsymbol{S}_{k-1} \tag{5.23}$$

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + 2\mu_k e_k \boldsymbol{S}_{k-1}\boldsymbol{x}_k \tag{5.24}$$

where

$$\mu_k = \frac{1}{2\tau_k} \tag{5.25}$$

The objective function of the KQN algorithm, on the other hand, assumes the form

$$J_{\boldsymbol{w}_k} = \sum_{i=1}^{k}[1-\mu_i]\left[\frac{d_i - \mu_i \boldsymbol{x}_i^T\boldsymbol{w}_{i-1}}{1-\mu_i} - \boldsymbol{x}_i^T\boldsymbol{w}_k\right]^2 + \frac{1}{2}\boldsymbol{w}_k^T\boldsymbol{R}_0\boldsymbol{w}_k \tag{5.26}$$

where $\mu_i = 1/(2\boldsymbol{x}_i^T\boldsymbol{S}_{i-1}\boldsymbol{x}_i)$. It turns out that in the KQN algorithm the value of $\mu_k - 1$ in the estimator in (5.23) approaches zero, as can be verified by examining Fig. 4 in [58]. As a result, the adaptation of $\boldsymbol{S}_k$ will stop after a certain number of iterations regardless of the value of $\|\boldsymbol{\delta}_k\|$ whereas the basic QN optimization algorithm as reported in [70] suggests that the adaptation of $\boldsymbol{S}_k$ should continue until the value of $\|\boldsymbol{\delta}_k\|$ becomes sufficiently small. Consequently, the steady-state value of the misalignment and the speed of convergence will be affected.

An unbiased estimate of the inverse of the input-signal autocorrelation matrix cannot be obtained by using the rank-one update formula given in (5.5). However, the undesired consequences of using a biased estimate in the adaptation of the weight vector can be avoided by using a convergence factor μ in (5.2) [3]. The autocorrelation matrix in (5.18) can be expressed as

$$\boldsymbol{R}_k = \boldsymbol{R}_0 + \sum_{i=1}^{k}\frac{\alpha_i}{\tau_{p,i}}\boldsymbol{x}_i\boldsymbol{x}_i^T \tag{5.27}$$

As can be seen, the update formula for $\boldsymbol{R}_k$ is a weighted sum of the outer product $\boldsymbol{x}_i\boldsymbol{x}_i^T$ and the weights depend on the input-signal statistics and the *a priori* error signal. Taking the expectation of both sides in (5.27) and invoking the assumption that τ and $\boldsymbol{x}_i\boldsymbol{x}_i^T$ are statistically independent, we obtain

$$E[\boldsymbol{R}_k] = \boldsymbol{R}_0 + \boldsymbol{R}_{x,x}\sum_{i=1}^{k}E\left(\frac{\alpha_i}{\tau_{p,i}}\right) \tag{5.28}$$

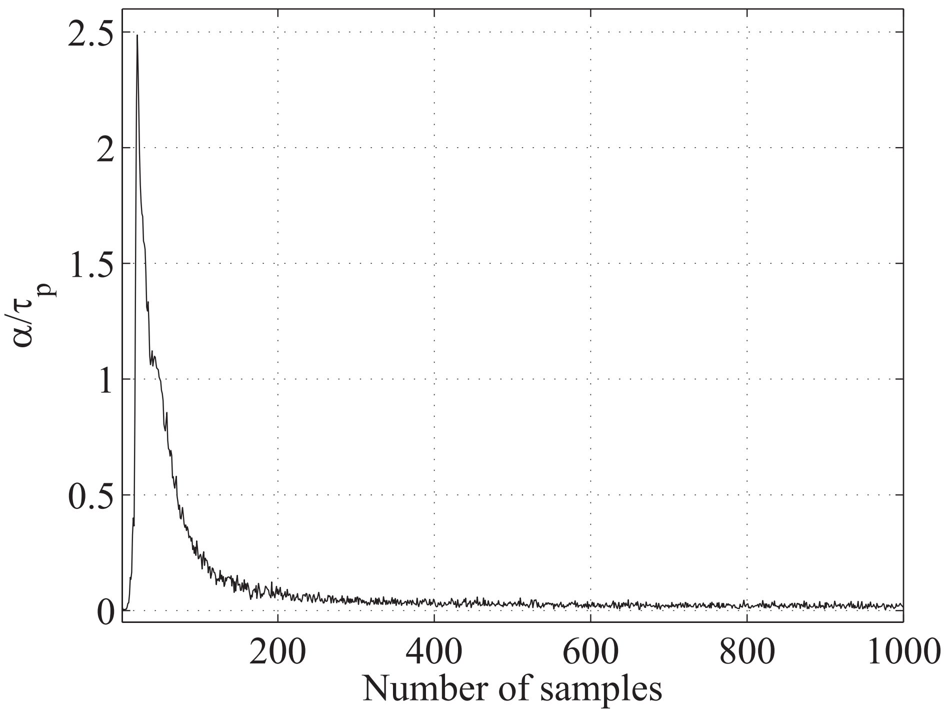

Figure 5.1: Evolution of α/τ_p.

An expression for the expectation of $\alpha_i/\tau_{p,i}$ is difficult to deduce but an approximate value can be obtained in terms of its time average based on simulation. In such an experiment, the error bound can be chosen as $\gamma = \sqrt{5\sigma_v^2}$ where σ_v^2 is the variance of the measurement noise. The evolution of α/τ_p for a white Gaussian input signal with zero mean and unit variance assuming a variance of the measurement noise, σ_v, of 10^{-4} is illustrated in Fig. 5.1. This was determined by averaging the ensemble of α/τ_p in 100 runs in a 36th-order system identification application.

Note that since $\alpha_i = 0$ for $|e_i| < \gamma$ and $\tau_{p,i} > 0$ as $\boldsymbol{S}_i$ is positive definite (see Theorem 1 below), we have $\alpha_i/\tau_{p,i} = 0$ for $|e_i| < \gamma$. Since $\boldsymbol{g}_k = \alpha_k \boldsymbol{t}_k$, the value of $\boldsymbol{t}_k$ does not need to be specified if $|e_k| < \gamma$.

As can be seen in Fig. 5.1, the time average is a positive quantity and, therefore, on the average a significant improvement can be brought about in the estimate of $\boldsymbol{R}_k$ with respect to $\boldsymbol{R}_0$. The effect of using a biased estimate on the weight vector will, therefore, be less pronounced in the proposed algorithm as the quantity α_k in the step size μ_k in (5.10) approaches zero at steady state. Since $\alpha_i \subset [0, 1)$ and $\tau_{p,i} > 0$ for all i, the weights used in (5.22) and (5.27) are nonnegative. Therefore, if $\boldsymbol{R}_0$ is positive definite it is straightforward to show that the estimate in (5.18), i.e., the Hessian of (5.1), would remain positive definite indefinitely.

The temporal behavior of α/τ_p in (5.22) can also be observed in Fig. 5.1. Since $\tau_{p,i} > 0$ and it is also bounded due to the fact that $\boldsymbol{S}_k$ is bounded (see Theorem 2 below), α/τ_p can approach zero only when $\alpha_i \approx 0$, i.e., when an approximate solution of the problem in (5.1) is achieved. During transience, $\alpha_i \approx 1$ and during steady state, $\alpha_i \approx 0$; therefore, α/τ_p is large during transience and becomes small at steady state.

The stability of Newton-type algorithms in general depends on the positive definiteness of $\boldsymbol{S}_k$ [58, 59]. Furthermore, $\boldsymbol{S}_k$ must be bounded for a bounded input signal. Otherwise, the bounded-input bounded-output (BIBO) stability of the algorithm cannot be assured. The formula in (5.16) could also lead to a biased solution for an unbounded $\boldsymbol{S}_k$. Both of these requirements are satisfied in the PQN algorithm according to the following theorems.

Theorem 1. *If $\boldsymbol{S}_{k-1}$ is a symmetric positive definite matrix, then $\boldsymbol{S}_k$ is also a symmetric positive definite matrix for all $k > 0$.*

Proof: Since $\boldsymbol{S}_{k-1}$ is a real symmetric matrix, we can express $\boldsymbol{S}_{k-1}$ as

$$\begin{aligned}\boldsymbol{S}_{k-1} &= \boldsymbol{U}\Lambda\boldsymbol{U}^T \\ &= \left(\boldsymbol{U}\Lambda^{1/2}\boldsymbol{U}^T\right)\left(\boldsymbol{U}\Lambda^{1/2}\boldsymbol{U}^T\right) \\ &= \boldsymbol{S}_{k-1}^{1/2}\boldsymbol{S}_{k-1}^{1/2}\end{aligned} \tag{5.29}$$

where $\boldsymbol{U}$ is a unitary matrix such that $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{I}$. If we let $\boldsymbol{u} = \boldsymbol{S}_{k-1}^{1/2}\boldsymbol{x}_k$ and $\boldsymbol{v} = \boldsymbol{S}_{k-1}^{1/2}\boldsymbol{z}$, then for any nonzero vector $\boldsymbol{z} \in \mathcal{R}^M$, the relation in (5.15) can be used to obtain

$$\begin{aligned}\boldsymbol{z}^T\boldsymbol{S}_k\boldsymbol{z} &= \boldsymbol{z}^T\boldsymbol{S}_{k-1}\boldsymbol{z} - \alpha_k \cdot \frac{\boldsymbol{z}^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{z}}{\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k} \\ &= \boldsymbol{z}^T\boldsymbol{S}_{k-1}\boldsymbol{z} - \alpha_k \cdot \frac{\left(\boldsymbol{z}^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k\right)^2}{\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k}\end{aligned}$$

Substituting (5.29) and the definitions of $\boldsymbol{u}$ and $\boldsymbol{v}$ in the above equation, we obtain

$$\begin{aligned}\boldsymbol{z}^T\boldsymbol{S}_k\boldsymbol{z} &= \boldsymbol{v}^T\boldsymbol{v} - \alpha_k \cdot \frac{\left(\boldsymbol{v}^T\boldsymbol{u}\right)^2}{\boldsymbol{u}^T\boldsymbol{u}} \\ &= \|\boldsymbol{v}\|^2 - \alpha_k \cdot \frac{\|\boldsymbol{v}\|^2\|\boldsymbol{u}\|^2\cos^2\theta}{\|\boldsymbol{u}\|^2} \\ &= \|\boldsymbol{v}\|^2 - \alpha_k \cdot \|\boldsymbol{v}\|^2(0.5 + 0.5\cos 2\theta)\end{aligned}$$

Since $0 < \alpha_k < 1$, the lowest possible value of the right-hand side in the above equation will occur when $\theta = 0$ and, therefore,

$$\boldsymbol{z}^T\boldsymbol{S}_k\boldsymbol{z} > \|\boldsymbol{v}\|^2(1 - \alpha_k) > 0 \tag{5.30}$$

for any $\boldsymbol{z} \in \mathcal{R}^{M\times 1}$. Hence the estimate in (5.15) is positive definite for all $k > 0$. The symmetry of $\boldsymbol{S}_k$ can be easily demonstrated by noting the symmetry of $\boldsymbol{S}_{k-1}$.

Theorem 2. *The estimate of $\boldsymbol{S}_k$ in (5.15) is always bounded in the sense that the quadratic factor $\boldsymbol{x}_k^T\boldsymbol{S}_k\boldsymbol{x}_k$ is bounded provided that the input signal is bounded.*

Proof: If we premultiply both sides in (5.15) by $\boldsymbol{x}_k^T$ and postmultiply them by $\boldsymbol{x}_k$, we obtain

$$\boldsymbol{x}_k^T\boldsymbol{S}_k\boldsymbol{x}_k = \boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k(1 - \alpha_k) \tag{5.31}$$

Since $0 < (1 - \alpha_k) < 1$ holds true for each adaptation, we have

$$\boldsymbol{x}_k^T\boldsymbol{S}_k\boldsymbol{x}_k < \boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k \tag{5.32}$$

Therefore, we conclude that if the input signal is bounded, then $\boldsymbol{S}_k$ is also bounded. Based on the above two theorems, the stability of the PQN algorithm can be established as discussed in the next section.

5.3 Stability Analysis

For purposes of analysis, we assume that the desired response for the adaptive filter is generated as

$$d_k = \boldsymbol{x}_k^T\boldsymbol{w}_{opt} \tag{5.33}$$

where $\boldsymbol{w}_{opt}$ is the weight vector of an FIR filter. We establish the convergence behavior of the proposed QN algorithm by examining the behavior of the weight-error vector defined as

$$\hat{\boldsymbol{w}}_k = \boldsymbol{w}_{opt} - \boldsymbol{w}_k \tag{5.34}$$

Using (5.33) and (5.34), the error signal in (5.3) can be expressed as

$$e_k = \boldsymbol{x}_k^T \hat{\boldsymbol{w}}_{k-1} \tag{5.35}$$

Subtracting $\boldsymbol{w}_{opt}$ from both sides of (5.16) and using the above relations, we obtain

$$\begin{aligned}\hat{\boldsymbol{w}}_k &= \left(\mathbf{I} - \alpha_k \frac{\boldsymbol{S}_{k-1}\boldsymbol{x}_k\boldsymbol{x}_k^T}{\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k}\right)\hat{\boldsymbol{w}}_{k-1} \\ &= \boldsymbol{P}_k \hat{\boldsymbol{w}}_{k-1} \end{aligned}\tag{5.36}$$

Global asymptotic convergence of the weight-error vector can be assured if and only if $E[\boldsymbol{P}_k]$ is time-invariant and its eigenvalues are strictly inside the unit circle. However, certain strong independence assumptions have to be made to obtain a time-invariant description of a system such as that represented by (5.36). As an alternative, conditions for convergence with probability 1 can be achieved using a system based on $\hat{\boldsymbol{w}}_k$ rather than $E[\hat{\boldsymbol{w}}_k]$. A similar approach has been used to demonstrate the stability of other algorithms, such as, for example, the known QN algorithm in [58] and the constrained affine projection algorithm in [27]. Since we have seen in Theorems 1 and 2 that the matrices $\boldsymbol{S}_k$ and $\boldsymbol{R}_k$ remain positive definite and bounded indefinitely , the following theorem can be established [58].

Theorem 3. *If the input signal is persistently exciting, then the system in (5.36) is stable and asymptotically stable and, consequently, the proposed QN algorithm is also stable and asymptotically stable.*

Proof. Since matrices $\boldsymbol{S}_k$ and $\boldsymbol{R}_k$ are bounded and invertible according to Lemma 1 in [58], an equivalent system of equations for $\hat{\boldsymbol{w}}_k$ in (5.36) (in the Lyapunov sense) can be obtained as

$$\overline{\hat{\boldsymbol{w}}}_k = \boldsymbol{R}_k^{1/2}\hat{\boldsymbol{w}}_k \tag{5.37}$$

where $\boldsymbol{R}_k = \boldsymbol{R}_k^{T/2}\boldsymbol{R}_k^{1/2}$. If the system represented by (5.37) is stable or unstable, then the system represented by (5.36) is also stable or unstable, as appropriate. Taking the Euclidian norm of both sides of (5.37), we obtain

$$\left\|\overline{\hat{\boldsymbol{w}}}_k\right\|^2 = \hat{\boldsymbol{w}}_k^T\boldsymbol{R}_k\hat{\boldsymbol{w}}_k \tag{5.38}$$

Substituting $\hat{\boldsymbol{w}}_k$ given by (5.36) in (5.38), we get

$$\left\|\overline{\hat{\boldsymbol{w}}}_k\right\|^2 = \hat{\boldsymbol{w}}_{k-1}^T \boldsymbol{P}_k^T \boldsymbol{R}_k \boldsymbol{P}_k \hat{\boldsymbol{w}}_{k-1} \tag{5.39}$$

On the other hand, substituting $\boldsymbol{R}_k$ given by (5.18) in (5.39), we have

$$\left\|\overline{\hat{\boldsymbol{w}}}_k\right\|^2 = \hat{\boldsymbol{w}}_{k-1}^T \boldsymbol{P}_k^T \boldsymbol{R}_{k-1} \boldsymbol{P}_k \hat{\boldsymbol{w}}_{k-1} + \frac{\alpha_k}{(1-\alpha_k)\tau_k} \cdot \hat{\boldsymbol{w}}_{k-1}^T \boldsymbol{P}_k^T \boldsymbol{x}_k \boldsymbol{x}_k^T \boldsymbol{P}_k \hat{\boldsymbol{w}}_{k-1} \tag{5.40}$$

It is easy to verify that

$$\boldsymbol{P}_k^T \boldsymbol{R}_{k-1} \boldsymbol{P}_k = \boldsymbol{R}_{k-1} - (2-\alpha_k)\frac{\alpha_k}{\tau_k} \boldsymbol{x}_k \boldsymbol{x}_k^T \tag{5.41}$$

and

$$\boldsymbol{P}_k^T \boldsymbol{x}_k \boldsymbol{x}_k^T \boldsymbol{P}_k = (1-\alpha_k)^2 \boldsymbol{x}_k \boldsymbol{x}_k^T \tag{5.42}$$

Now if we use (5.41) and (5.42) in (5.40) and then use (5.38), we have

$$\left\|\overline{\hat{\boldsymbol{w}}}_k\right\|^2 = \left\|\overline{\hat{\boldsymbol{w}}}_{k-1}\right\|^2 - \frac{\alpha_k}{\tau_k} \hat{\boldsymbol{w}}_{k-1}^T \boldsymbol{x}_k \boldsymbol{x}_k^T \hat{\boldsymbol{w}}_{k-1} \tag{5.43}$$

Since $\alpha_k \subset (0,1)$ and $\tau_k > 0$ for all k, we obtain

$$\left\|\overline{\hat{\boldsymbol{w}}}_k\right\|^2 \leq \left\|\overline{\hat{\boldsymbol{w}}}_{k-1}\right\|^2 \tag{5.44}$$

The left-hand side in (5.44) would be equal to the right-hand side in an interval $[k_1, k_2]$ if and only if $\boldsymbol{x}_k$ remains orthogonal to $\hat{\boldsymbol{w}}_{k-1}$ for all $k \in [k_1, k_2]$. However, in such a situation we would obtain $\hat{\boldsymbol{w}}_k = \hat{\boldsymbol{w}}_{k-1}$ for all $k \in [k_1, k_2]$ in (5.36). Therefore, $\left\|\overline{\hat{\boldsymbol{w}}}_k\right\|^2 = \left\|\overline{\hat{\boldsymbol{w}}}_{k-1}\right\|^2$ would hold true if and only if $\overline{\hat{\boldsymbol{w}}}_{k_1} = \overline{\hat{\boldsymbol{w}}}_{k_1+1} = \cdots = \overline{\hat{\boldsymbol{w}}}_{k_2} = \overline{\hat{\boldsymbol{w}}}$. However, if the input signal is persistently exciting, then there is an infinite number of sets $S_i = \{\boldsymbol{x}_{k_{1,i}}, \ldots \boldsymbol{x}_{k_{2,i}}\}$ with $M \leq k_{2,i} - k_{1,i} < M'$ such that each set S_i completely spans $\Re^M$ for some finite value of $M' > 0$ [58, 35]. Thus it would be impossible for $\hat{\boldsymbol{w}}_{k-1}$ to be orthogonal to $\boldsymbol{x}_k$ for all $\boldsymbol{x}_k \in S_i$ and as a result $\left\|\overline{\hat{\boldsymbol{w}}}_k\right\|^2 < \left\|\overline{\hat{\boldsymbol{w}}}_{k-1}\right\|^2$ which proves that the system in (5.36) and, in turn, the proposed QN algorithm is stable. As the number of sets is infinite, it follows that $\left\|\hat{\boldsymbol{w}}_k\right\|^2 \to 0$ for $k \to \infty$ and, therefore, the system in (5.36) and the proposed QN algorithm are stable and asymptotically stable. □

5.4 MSE Analysis of Proposed QN Algorithm

In this section, we examine the performance of the PQN algorithm in terms of the mean square-error (MSE) after initial convergence in stationary and nonstationary environments. The update formula for the weight vector in the PQN algorithm is similar to that of the LMS-Newton algorithm given in Eq. (29) in [59]. The difference resides in the estimation of the inverse of the autocorrelation matrix and the *reduction factor*, q. The KQN algorithm uses $q = 1$ instead of a prespecified fixed reduction factor q and the PQN algorithm uses a variable reduction factor $q = \alpha_k = [0, 1)$. However, the steady-state MSE of the PQN algorithm depends on the steady-state value of $\alpha_k = p$, not on its transient value. As reported in [59], the steady-state mean-square error given in Eqs. (40) and (46) of [59] for stationary and nonstationary environments, respectively, is independent of the way the inverse of the autocorrelation matrix is estimated and hence it will not be different for other Newton-type algorithms as long as (1) they use a weight-vector update equation of the type given in [59], (2) they use an approximate Newton direction, and (3) the assumptions made in [59] hold true. This conclusion is corroborated in [58] where the expression for the steady-state MSE in the KQN algorithm is shown to be identical with that of the LMSN algorithms. As can be verified, using $q = 1$ in Eq. (46) of [59], Eq. (22) of [58] can be obtained. Since the PQN algorithm follows an approximate Newton direction, employs a similar step size, and uses the same update equations for the weight vector, the formulas for the excess mean-square error are the same as those in Eqs. (40) and (46) in [59]. For stationary environments, the excess mean-square error for the PQN algorithm is given by

$$J_{ex,PQN} = \frac{p}{2-p} J_{min} \tag{5.45}$$

where J_{min} is the minimum mean-squared error and p is the value of α_k as $k \to \infty$. For the case of the KQN algorithm, this becomes $J_{ex,QN} = J_{min}$. As $p \ll 1$ at steady state, we have $J_{ex,PQN} < J_{ex,KQN}$. In addition, we have $J_{ex,PQN} < J_{ex,LMS-Newton}$ for a prespecified q.

If the weights of the unknown plant change according to the update formula

$$\boldsymbol{w}_{o,k} = \boldsymbol{w}_{o,k-1} + \nu_k \tag{5.46}$$

the excess mean-square error is given by

$$J_{ex,PQN} = \frac{1}{2-p}\left(pJ_{min} + \frac{\sigma_x^2\sigma_\nu^2 M^2}{p}\right) \tag{5.47}$$

where σ_x^2 is the variance of the input signal and σ_ν^2 is the variance of the elements of ν. As can be seen, the second term in the parenthesis is inversely proportional to p and, therefore, α_k should not be allowed to become too small.

The optimal value of the reduction factor, q_o, that minimizes the excess mean-square error is given in Eq. (47) of [59]. As can be easily verified, it is difficult to determine the optimal reduction factor as parameter a in Eq. (47) of [59] is unknown *a priori*. Since the derivation of q_o involves certain assumptions, there is no guarantee that the minimum excess mean-square error will be obtained with q_o [59]. To circumvent these problems, a small positive constant ϵ can be added to α_k to be used in the weight update formula for nonstationary environments so that $\alpha_{k\to\infty} \approx \epsilon$.

A numerical value for p is difficult to obtain if the input signal is colored. However, for a white Gaussian input signal an approximate range of p can be obtained as

$$2Q\left(\frac{\gamma}{\sigma_v}\right) \le p \le 2Q\left(\frac{\gamma}{\sqrt{2\sigma_v^2+\gamma^2}}\right) \tag{5.48}$$

where σ_v^2 is the variance of the noise signal added to the desired signal and $Q(\cdot)$ is the complementary Gaussian cumulative distribution function [35]. The maximum reduction in the number of weight updates can be obtained by using the Chebyshev equality

$$Pr(|e_k| < \gamma)_{max} = 1 - Pr(|e_k| > \gamma) = 1 - \frac{\sigma_v^2}{\gamma^2} \tag{5.49}$$

where σ_v^2 is the minimum value of σ_e^2 and γ^2 is chosen as an integer multiple of σ_v^2, i.e., $\gamma^2 = k\sigma_v^2$ with $k = 1, 2, \ldots, L$.

The number of updates, convergence speed, and steady-state MSE depend on the value of parameter γ. From (5.49), a larger value of γ would reduce the number of updates. From (5.48), on the other hand, we note that a larger γ would reduce p, and according to (5.45) a reduced steady-state misalignment would be achieved in stationary environments. However, in such a case the convergence speed of the algorithm would be compromised. A smaller value of γ, on the other hand, would increase the number of updates and p and, therefore, an increased steady-state misalignment would result in the case of stationary environments. The convergence speed of the algorithm in this case would be improved. Similar conclusions about the influence of γ on the number of updates, convergence speed, and steady-state MSE were also drawn in [35]. However, such conclusions cannot be deduced for nonstationary environments as the relation in (5.47) is nonlinear. In nonstationary environments, a reduced error bound would improve the tracking capability of the algorithm because the algorithm would continue to carry out updates after convergence. Experimentation has shown that good performance can be achieved in stationary and nonstationary environments by choosing integer k in γ in the range of 1 to 5 in the first case and 1 to 3 in the second case.

As far as stability is concerned, the proposed algorithm is inherently stable as the *a posteriori* error is forced to be equal to the prespecified error bound, γ, whenever an update is made. A rough approximation of the variance of the measurement noise would be enough to choose the error bound. In certain engineering applications, the measurement noise has an upper bound [100] and in such applications the PQN algorithm can be readily applied.

5.5 Simulation Results

In order to evaluate the performance of the proposed QN algorithm several experiments were carried out as detailed below.

In the first experiment, an adaptive filter was used to identify a 36th-order lowpass FIR filter with a cutoff frequency of $0.3\omega_s$, where ω_s is the sampling frequency, using normalized coefficients to assure that the total power is unity. The input signal was a sinusoid of amplitude 1 and frequency of $\omega_s/16$ and was contaminated by a Gaussian noise signal of zero mean and variance 0.1. The contaminating Gaussian noise signal was colored using a 10th-order lowpass FIR filter with a cutoff frequency of $0.5\omega_s$. A sinusoidal signal was chosen because it causes the input signal to be severely ill-conditioned. With such a system identification problem, the convergence behavior of Newton-type algorithms can be better understood as their convergence speed would be low enough to facilitate comparison. The measurement noise added to the desired signal was white Gaussian with zero mean and variance of $10^{-2},\ 10^{-6},\ 10^{-10},$ and 10^{-14} to achieve signal-to-noise ratios (SNRs) of $20, 60, 100,$ and 140 dB, respectively. The prespecified error bound was chosen as $\gamma = \sqrt{5\sigma_v^2}$ where σ_v^2 is the variance of the measurement noise. The initial weight vector was assumed to be the zero vector and the estimate of the inverse of the

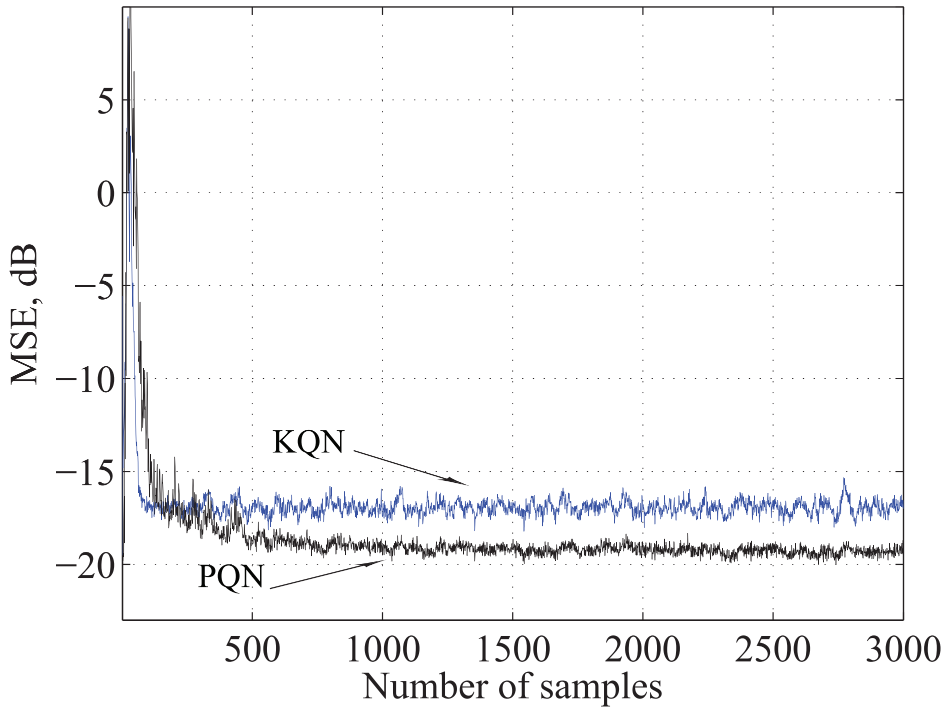

Figure 5.2: Learning curves in a stationary environment with SNR = 20 dB.

autocorrelation matrix was assumed to be the identity matrix in all experiments in the PQN as well as the KQN algorithms. The tolerance factor ς used in the fixed-point implementations was $\varsigma = 10^{-3}$ [57]. The learning curves obtained for different SNRs from 1000 independent trials by using the PQN and the KQN algorithms in a stationary environment are illustrated Figs. 5.2–5.5.

The number of iterations required for convergence, the steady-state misalignment, the number of weight updates required by the PQN and KQN algorithms in the above experiment in 3000 iterations, and the reductions in the number of updates achieved are given in Table 5.2.

The second experiment was identical to the first experiment except that a nonstationarity was introduced in the filter taps $\boldsymbol{h}_k$ according to the first-order Markov model

$$\boldsymbol{h}_k = \boldsymbol{h}_{k-1} + \boldsymbol{\nu}_k \tag{5.50}$$

where the entries of $\boldsymbol{\nu}$ were the samples of a white Gaussian noise sequence with zero mean and variance equal to $10^{-8}, 10^{-12}, 10^{-14}$, and 10^{-18}. The prespecified error bound in nonstationary environments was chosen as $\gamma = \sqrt{3\sigma_v^2}$. With a smaller error

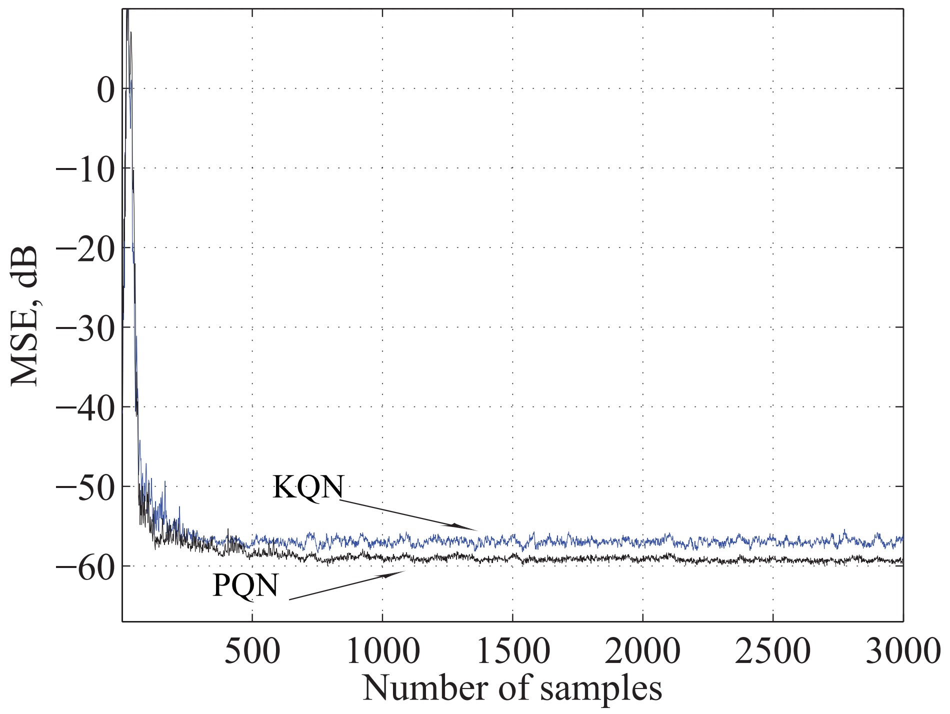

Figure 5.3: Learning curves in a stationary environment with SNR = 60 dB.

bound, the number of adaptations is increased and, therefore, the tracking of the changes in $\boldsymbol{h}_k$ as given in (5.50) improves after reaching steady state. The learning curves for different SNRs obtained from 1000 independent trials by using the PQN and KQN algorithms in a nonstationary environment are illustrated Figs. 5.6–5.9.

The number of iterations to converge, the steady-state misalignment, and the number of weight updates required by the PQN and KQN algorithms in 3000 iterations and the reductions achieved are given in Table 5.3.

As can be seen in Figs. 5.2–5.5 and Figs. 5.6–5.9 and Tables 5.2 and 5.3, the PQN algorithm yields reduced misalignment while requiring fewer iterations to converge than the KQN algorithm for stationary and nonstationary environments. As in the KQN algorithm, the learning curves at steady-state in the PQN algorithm are not noisy. The improvement in the steady-state misalignment becomes more prominent for high SNRs in the PQN algorithm because in the KQN algorithm adaptation of the inverse of the autocorrelation matrix stops prior to reaching steady state. Tables 5.2 and 5.3 also show that the required numbers of weight updates in the PQN algorithm are only a fraction of those required by the KQN algorithm.

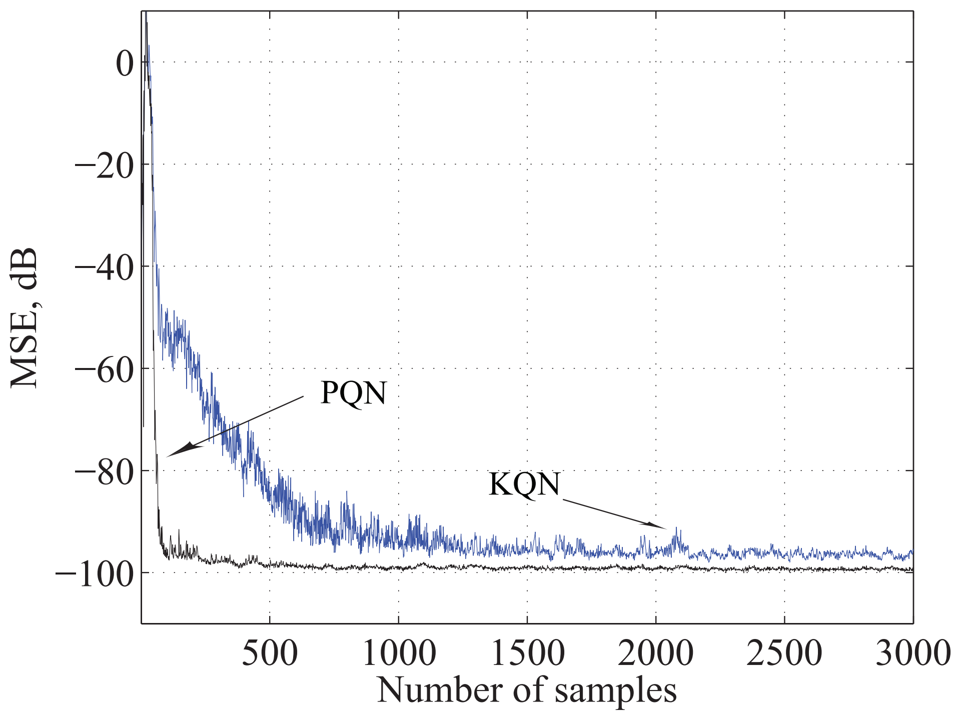

Figure 5.4: Learning curves in a stationary environment with SNR = 100 dB.

In the third and fourth experiments, we verified the formulas in (5.45) and (5.47) for the excess MSE for different system orders and different values of the error bound. The input signal was white Gaussian noise with zero mean and variance 0.1. The limiting values of p for the error bound $\gamma = \sqrt{k\sigma_v^2}$ for $k = 1, \ldots, 10$, were obtained using (5.48). The results presented in Tables 5.4–5.7 are the outcome of an ensemble of 100 runs where p_{min} and p_{max} are the limiting values of p and σ_v^2 and M denote the variance of the measurement noise and system order, respectively. As can be seen in Tables 5.4–5.7, the excess MSE obtained from simulation lies within the range of the excess MSE obtained by using p_{min} and p_{max} in (5.45) for stationary environments as expected. In addition, the excess MSE reduces as the error bound is increased as discussed in the Sec. 6.4.

The fourth experiment was the same as the third except that a nonstationarity was introduced as in the second experiment. The results obtained are given in Tables 5.8–5.11 where σ_ν^2 is the variance of the Gaussian noise added to the weight vector. In nonstationary environments, the excess MSE obtained by simulation may not be within the range of the excess MSE obtained by using (5.47) since this formula remains

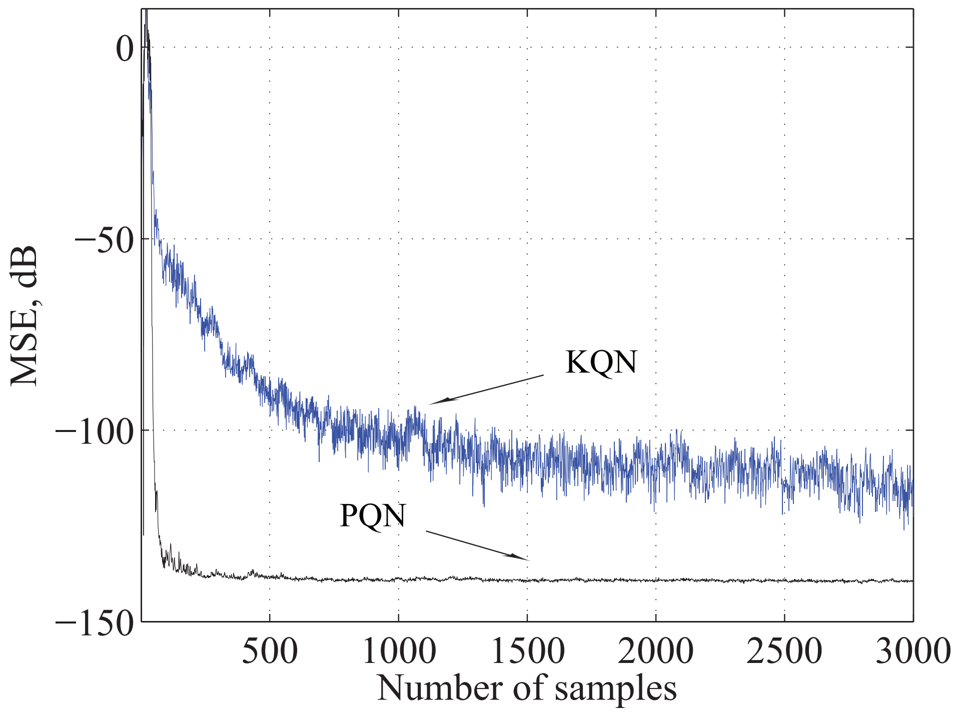

Figure 5.5: Learning curves in a stationary environment with SNR = 140 dB.

nonlinear at steady state. However, the simulation results obtained are very close to the theoretical results.

The final experiment was carried out to demonstrate the robustness of the proposed algorithm in the case of a fixed-point implementation. The system to be identified was the same as that used in the first experiment. Here the input signal was a zero-mean white Gaussian noise with a variance of unity and was colored using an 10th-order lowpass FIR filter with a cutoff frequency of $0.5\omega_s$. The error bound was chosen as $\gamma = \sqrt{5\sigma_v^2}$ where σ_v^2 is the variance of the measurement noise. Fixed-point arithmetic was assumed using a word length of 20 bits with no scaling or rescuing procedures and overflow was handled using saturation arithmetic. The error signal, error bound, and the desired signal were quantized and the learning curves were not smoothed. The learning curves obtained by using the PQN and KQN algorithms in 100 trials with $\sigma_v^2 = 10^{-4}$ and 10^{-8}, and SNRs of 40 and 80 dB are illustrated in Fig. 5.10. These results are consistent with the results obtained with floating-point arithmetic. Furthermore, Fig. 5.10 shows that when implemented in fixed-point arithmetic the PQN algorithm is as robust as the KQN algorithm.

Table 5.2: Comparison of Proposed with Known QN Algorithm

SNR	20 dB		60 dB		100 dB		140 dB	
Algorithm	PQN	KQN	PQN	KQN	PQN	KQN	PQN	KQN
Iterations to converge	120	120	130	130	120	700	100	3000
Weight updates	207	3000	229	3000	225	3000	228	3000
Reduction, %	93	-	92	-	92	-	92	-
Steady-state misalignment, dB	-19.3	-16.9	-59.3	-56.9	-99.3	-96.6	-139.4	-114.8

Table 5.3: Comparison of Proposed with Known QN Algorithm

SNR	20 dB		60 dB		100 dB		140 dB	
Algorithm	PQN	KQN	PQN	KQN	PQN	KQN	PQN	KQN
Iterations to converge	120	120	130	200	130	860	200	3000
Weight updates	505	3000	450	3000	685	3000	874	3000
Reduction, %	83	-	85	-	77	-	71	-
Steady-state misalignment, dB	-18.4	-16.7	-59.0	-56.8	-97.3	-96.0	-137.5	-118.3

5.6 Conclusions

An improved QN adaptation algorithm that incorporates data selective adaptation for the weight vector and the inverse of the autocorrelation matrix has been proposed. The proposed algorithm was developed on the basis of the framework of classical QN optimization algorithms and in this way an improved estimator of the autocorrelation matrix was deduced. Analysis has shown that the PQN algorithm should perform better than the KQN algorithm in terms of convergence speed and steady-state misalignment. This expectation has been substantiated by simulations that have shown that when compared with the KQN algorithm, the PQN algorithm requires fewer iterations to converge, fewer weight updates, and yields reduced steady-state misalignment in stationary as well as nonstationary environments. In addition, analytical results obtained by using closed-form formulas for the steady-state misalignment agree well with those obtained by simulations. The PQN algorithm was also found to be as robust as the KQN algorithm in the case of fixed-point implementations.

In the next chapter, we describe a robust version of the PQN algorithm for applications that entail impulsive noise.

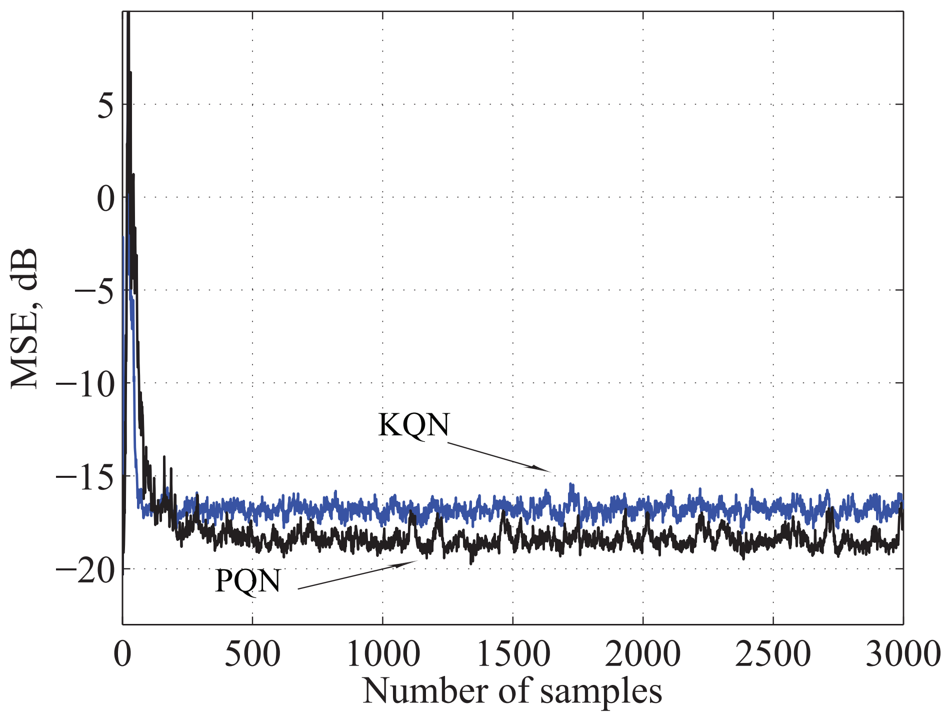

Figure 5.6: Learning curves in a nonstationary environment with SNR = 20 dB.

Table 5.4: Excess MSE in dB in Proposed QN Algorithm
$M = 36,\ \sigma_v^2 = 10^{-4}$

Error bound k	Eq. (5.45) p_{min}	Simulation	Eq. (5.45) p_{max}
1	−39.2	−38.4	−38.5
2	−39.6	−39.0	−38.8
3	−39.8	−39.2	−38.9
4	−39.9	−39.5	−38.9
5	−39.9	−39.6	−39.0
6	−39.9	−39.7	−39.0
7	−39.9	−39.6	−39.0
8	−39.9	−39.5	−39.1
9	−39.9	−39.4	−39.1
10	−39.9	−39.2	−39.1

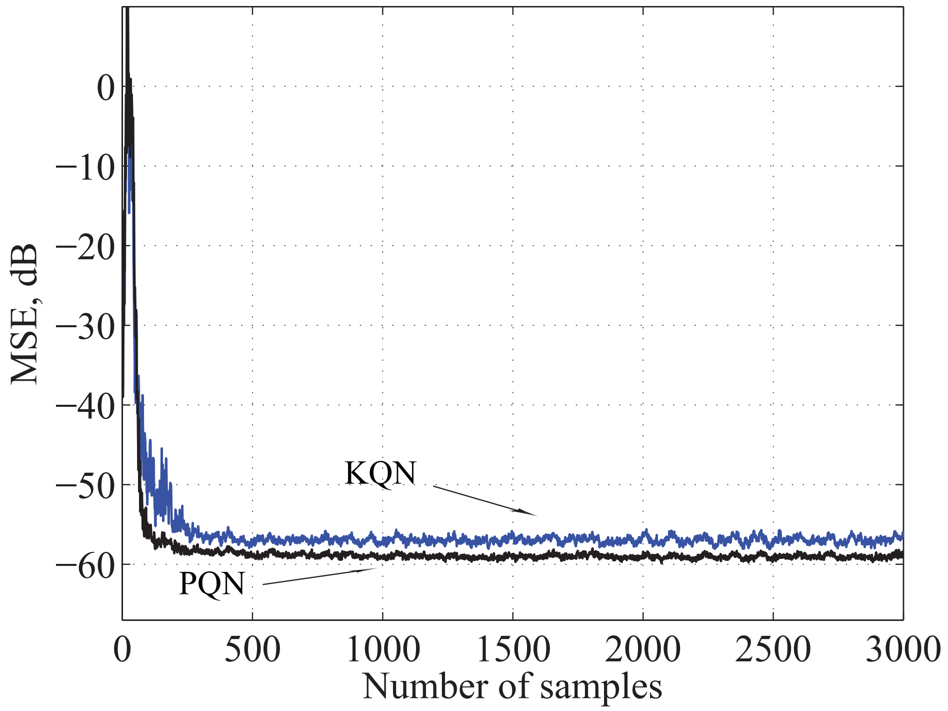

Figure 5.7: Learning curves in a nonstationary environment with SNR = 60 dB.

Table 5.5: Excess MSE in dB in Proposed QN Algorithm
$M = 36,\ \sigma_v^2 = 10^{-8}$

Error bound	Eq. (5.45)	Simulation	Eq. (5.45)
k	p_{min}		p_{max}
1	−79.2	−78.2	−78.5
2	−79.6	−78.7	−78.8
3	−79.8	−79.0	−78.9
4	−79.9	−79.2	−78.9
5	−79.9	−79.3	−79.0
6	−79.9	−79.5	−79.0
7	−79.9	−79.5	−79.0
8	−79.9	−79.4	−79.1
9	−79.9	−79.2	−79.1
10	−79.9	−79.0	−79.1

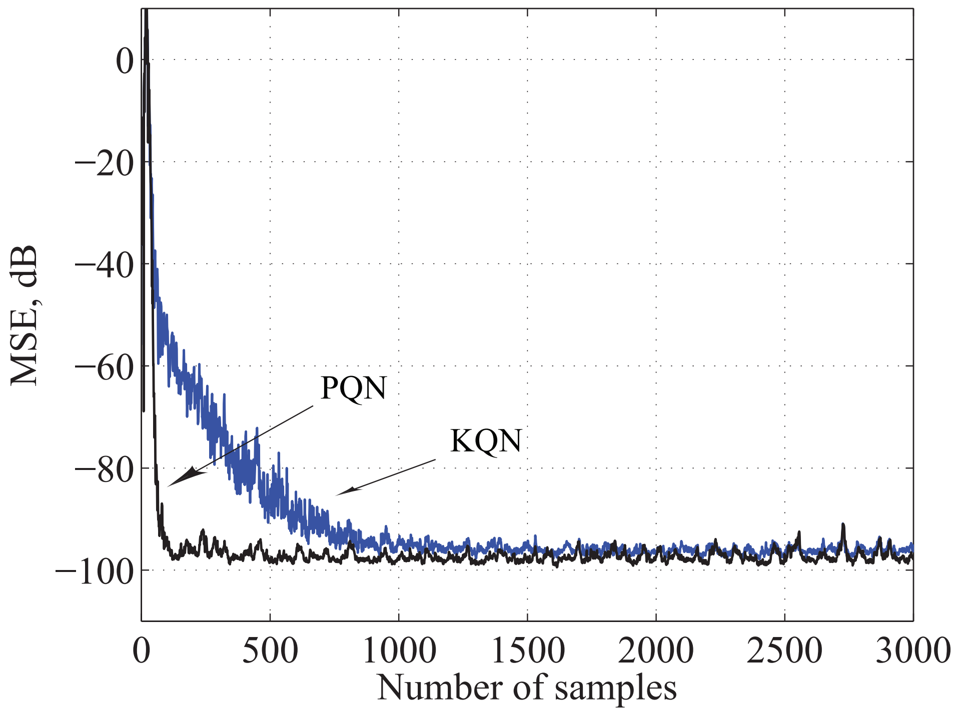

Figure 5.8: Learning curves in a nonstationary environment with SNR = 100 dB.

Table 5.6: Excess MSE in dB in Proposed QN Algorithm
$M = 56,\ \sigma_v^2 = 10^{-4}$

Error bound k	Eq. (5.45) p_{min}	Simulation	Eq. (5.45) p_{max}
1	−39.2	−38.4	−38.5
2	−39.6	−38.9	−38.8
3	−39.8	−39.1	−38.9
4	−39.9	−39.2	−38.9
5	−39.9	−39.2	−39.0
6	−39.9	−39.2	−39.0
7	−39.9	−39.1	−39.0
8	−39.9	−39.1	−39.1
9	−39.9	−38.9	−39.1
10	−39.9	−38.7	−39.1

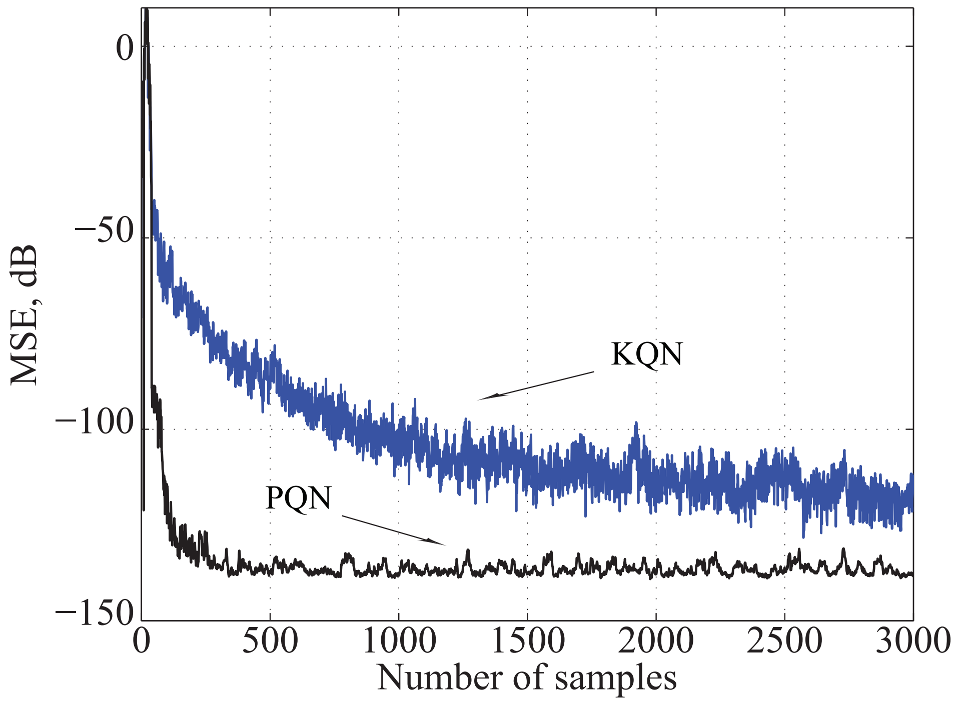

Figure 5.9: Learning curves in a nonstationary environment with SNR = 140 dB.

Table 5.7: Excess MSE in dB in Proposed QN Algorithm
$M = 56,\ \sigma_v^2 = 10^{-8}$

Error bound k	Eq. (5.45) p_{min}	Simulation	Eq. (5.45) p_{max}
1	−79.2	−78.2	−78.5
2	−79.6	−78.8	−78.8
3	−79.8	−79.1	−78.9
4	−79.9	−79.3	−78.9
5	−79.9	−79.4	−79.0
6	−79.9	−79.5	−79.0
7	−79.9	−79.4	−79.0
8	−79.9	−79.2	−79.1
9	−79.9	−79.0	−79.1
10	−79.9	−78.8	−79.1

Table 5.8: Excess MSE in dB in Proposed QN Algorithm
$M = 36,\ \sigma_v^2 = 10^{-4},\ \sigma_\nu^2 = 10^{-8}$

Error bound k	Eq. (5.47) p_{min}	Simulation	Eq. (5.47) p_{max}
1	−37.3	−37.8	−37.6
2	−37.0	−38.1	−37.5
3	−36.8	−38.1	−37.5
4	−36.8	−38.1	−37.4
5	−36.7	−38.0	−37.4
6	−36.7	−37.8	−37.4
7	−36.7	−37.6	−37.4
8	−36.7	−37.4	−37.3
9	−36.7	−37.2	−37.3
10	−36.7	−37.0	−37.3

Table 5.9: Excess MSE in dB in Proposed QN Algorithm
$M = 36,\ \sigma_v^2 = 10^{-8},\ \sigma_\nu^2 = 10^{-14}$

Error bound k	Eq. (5.47) p_{min}	Simulation	Eq. (5.47) p_{max}
1	−77.6	−78.2	−78.0
2	−77.3	−78.7	−77.9
3	−77.1	−78.9	−77.8
4	−77.0	−79.1	−77.8
5	−77.0	−79.2	−77.7
6	−77.0	−79.3	−77.7
7	−77.0	−79.3	−77.7
8	−76.9	−79.3	−77.7
9	−76.9	−79.1	−77.7
10	−76.9	−79.0	−77.7

Table 5.10: Excess MSE in dB in Proposed QN Algorithm
$M = 56,\ \sigma_v^2 = 10^{-4},\ \sigma_\nu^2 = 10^{-8}$

Error bound	Eq. (5.47)	Simulation	Eq. (5.47)
k	p_{min}		p_{max}
1	−36.8	−37.1	−37.2
2	−36.6	−37.1	−37.1
3	−36.5	−37.1	−37.0
4	−36.4	−37.0	−37.0
5	−36.4	−36.9	−37.0
6	−36.3	−36.7	−36.9
7	−36.3	−36.7	−36.9
8	−36.3	−36.6	−36.9
9	−36.3	−36.2	−36.9
10	−36.3	−36.0	−36.9

Table 5.11: Excess MSE in dB in Proposed QN Algorithm
$M = 56,\ \sigma_v^2 = 10^{-8},\ \sigma_\nu^2 = 10^{-14}$

Error bound	Eq. (5.47)	Simulation	Eq. (5.47)
k	p_{min}		p_{max}
1	−77.6	−78.3	−78.0
2	−77.3	−78.8	−77.9
3	−77.1	−79.0	−77.8
4	−77.0	−79.2	−77.8
5	−77.0	−79.2	−77.7
6	−77.0	−79.3	−77.7
7	−77.0	−79.2	−77.7
8	−76.9	−79.1	−77.7
9	−76.9	−78.9	−77.7
10	−76.9	−78.7	−77.7

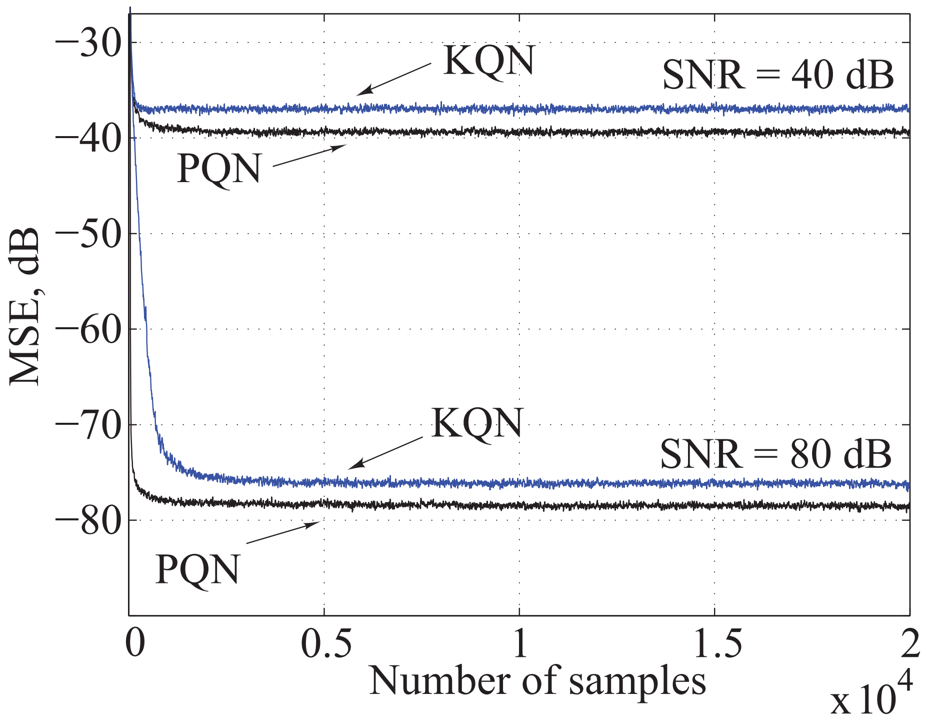

Figure 5.10: Learning curve for a fixed-point implementation.

Chapter 6

Robust Quasi-Newton Adaptation Algorithms

6.1 Introduction

Both of the known quasi-Newton (KQN) algorithms proposed in [58] and the proposed QN (PQN) algorithm in Table 5.1 are not robust with respect to impulsive noise. In this chapter, we propose two new robust QN (RQN) algorithms that are robust with respect to impulsive noise and perform data selective adaptation in updating the inverse of the autocorrelation matrix and the weight vector [86]. The new algorithms are essentially enhancements of the algorithms described in chapter 5 [85] for applications that entail impulsive noise. Like the RSMAP and PCSMAP algorithms, the proposed RQN algorithms use two error bounds. One of the error bounds yields faster convergence and good re-adaptation capability whereas the other reduces the influence of impulsive noise and yields a reduced steady-state misalignment without affecting the convergence speed. Switching between the two error bounds is achieved by comparing the absolute value of the error signal with a threshold. A simple variance estimator for the error signal is used to determine the threshold. The new algorithms are also asymptotically stable. A mean-square error (MSE) analysis based on the recently proposed energy conservation relation is carried out. Simulation results concerning the steady-state MSE in a system identification application match quite well the analytical steady-state MSE. Furthermore, simulation results show that the proposed algorithms offer improved performance relative to the KQN [58] and robust QN (RQN) [64] algorithms in terms of convergence speed, re-adaptation capability, computational efficiency, and robustness with respect to impulsive noise.

6.2 Proposed Robust Quasi-Newton Algorithms

Two slightly different RQN algorithms are possible one using a fixed threshold and the other using a variable threshold as detailed below.

6.2.1 RQN Algorithm with Fixed Threshold

The update equations for matrix $\boldsymbol{S}_k$ and the corresponding weight vector $\boldsymbol{w}_k$ for the PQN algorithm given by (5.15) and (5.16), respectively, are rewritten here for convenience as

$$\boldsymbol{S}_k = \boldsymbol{S}_{k-1} - \alpha_k \frac{\boldsymbol{S}_{k-1}\boldsymbol{x}_k\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}}{\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k} \tag{6.1}$$

and

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \alpha_k \frac{\boldsymbol{S}_{k-1}\boldsymbol{x}_k}{\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k} e_k \tag{6.2}$$

where $\boldsymbol{x}_k$ is the input-signal vector and

$$\alpha_k = \begin{cases} 1 - \dfrac{\gamma}{|e_k|} & \text{if } |e_k| > \gamma \\ 0 & \text{otherwise} \end{cases} \tag{6.3}$$

Parameter γ in (6.3) is the prespecified error bound and e_k is the noisy *a priori* error signal given by

$$e_k = d_k - \boldsymbol{w}_{k-1}^T\boldsymbol{x}_k \tag{6.4}$$

In order to achieve robust performance against impulsive noise, we choose this error bound as

$$\gamma = \begin{cases} |e_k| - \nu\theta_k & \text{if } |e_k| > \theta_k \\ \gamma_c & \text{otherwise} \end{cases} \tag{6.5}$$

where γ_c is a prespecified error bound chosen as $\sqrt{5}\sigma_v$ where σ_v^2 is the variance of the measurement noise, ν such that $0 < \nu \ll 1$ is a scalar, and θ_k is a threshold parameter which is estimated as

$$\theta_k = 1.98\sigma_k \tag{6.6}$$

where

$$\sigma_k^2 = \lambda\sigma_{k-1}^2 + (1-\lambda)\min(\boldsymbol{g}_k) \tag{6.7}$$

with $\lambda \subset (0,1)$, $\boldsymbol{g}_k = [e_k^2+\epsilon \ \cdots \ e_{k-P+1}^2+\epsilon]$ is a vector of size P, and ϵ is a small scalar. Whenever $|e_k| < \gamma$, no update is applied. Consequently, the amount of computation as well as the required storage are significantly reduced since $\boldsymbol{S}_k$ is not evaluated in every iteration. A similar weight vector update strategy has been used in set-membership adaptation algorithms [19, 35, 27] but the mathematical framework of those algorithms is very different from that of the proposed RQN algorithm. The estimator in (6.7) is robust to outliers. Under the assumption that the probability distribution of e_k is Gaussian, the above choice of θ_k would ensure that impulsive noise can be suppressed with a probability of just over 0.95. A large σ_0^2 would cause $|e_k|$ to be less than θ_k during the transient state and, therefore, the algorithm would work with error bound γ_c which would increase the initial rate of convergence. For a sudden system disturbance, θ_k would also be very large in which case we obtain $\gamma_c < |e_k| < \theta_k$ and thus the algorithm would again use error bound γ_c and, therefore, the tracking capability of the algorithm would be retained. For an impulsive noise-corrupted error signal, θ_k would not increase in which case the error bound would be $\theta_k = |e_k| - \nu\theta_k$ and this would suppress the impulsive noise.

6.2.2 RQN Algorithm with Variable Threshold

The RQN algorithm with variable threshold is essentially the same as the RQN algorithm with fixed threshold except that the error bound γ_c in (6.5) is estimated as

$$\xi_k = \beta\xi_{k-1} + (1-\beta)\min\left(\xi_{k-1}, \frac{|d_k^2 - y_k^2|}{d_k^2}\right) \tag{6.8}$$

$$\hat{\sigma}_k^2 = \beta\hat{\sigma}_{k-1}^2 + (1-\beta)\min(\hat{\sigma}_{k-1}^2, \sigma_k^2) \tag{6.9}$$

$$\gamma_{c,k} = \sqrt{\xi_k\gamma_{c,0}} + 1.12\,[1+\text{sign}(1-\xi_k)]\,\hat{\sigma}_k \tag{6.10}$$

where $\gamma_{c,0}$ is a rough estimate of σ_v and $\xi_0 \gg 1$. During steady state, we obtain $\xi_k \approx 0$ and $\hat{\sigma}_k^2 \approx \sigma_v^2$ and hence $\gamma_{c,k}$ would be $2.24\sigma_v$, thereby resulting in reduced steady-state misalignment. In the transient state, $\xi_k \approx \xi_0$ and, therefore, the algorithm would use $\gamma_{c,k} = \sqrt{\xi_k\gamma_{c,0}}$ and this would yield faster convergence. The parameters in (6.8) and (6.9) are robust to outliers as each is based on the minimum value of its two most recent values. On the other hand, the variance is estimated as

$$\sigma_k^2 = \lambda\sigma_{k-1}^2 + (1-\lambda)\text{median}(\boldsymbol{g}_k) \tag{6.11}$$

For Gaussian signals, the median of the squares of the signal samples in the variance estimator usually gives a more accurate estimate of the variance of the signal than the instantaneous values of the squares of the signal samples. The use of the median operation in adaptive filters was introduced by Zou et al. in [63] and was later used by other researchers, e.g., in [68].

A variable threshold γ_c is useful in applications where the noise variance σ_v^2 is unknown.

6.2.3 Discussion

In the proposed algorithms, we obtain $0 \leq \alpha_k < 1$ for both values of γ in (6.5) and hence the estimate in (5.15) would remain positive definite indefinitely if $\boldsymbol{S}_k$ is initialized with a positive definite matrix [85]. The RLS-type robust algorithms in [67] and [68] are implemented using a fast transversal filter implementation [3] to reduce their computational complexity from order M^2 denoted as $\mathcal{O}(M^2)$ to $\mathcal{O}(M)$ and, therefore, both algorithms would inherit the problems associated with the RLS algorithms of order M^2. Moreover, the Huber function used in [67] does not have a closed-form solution and hence the solution obtained can be suboptimal and, in addition, its tracking capability can be compromised [68]. The solution obtained in [68] can also be suboptimal [68]. The adaptation algorithm in [68] is robust in the sense that it returns to the true solution without losing its initial convergence speed after being subjected to impulsive noise. The proposed RQN algorithms are also robust with respect to impulsive noise in the sense that they return to the true solution faster than the initial convergence. The known QN and RQN algorithms do not employ a variable step size. However, the known RQN algorithm requires an additional amount of computation of $\mathcal{O}(M\log_2(M))$ per iteration as compared to the known QN algorithm and the proposed QN algorithm with fixed threshold.

The known RQN algorithm in [64] uses the update equations

$$\boldsymbol{S}_k = \lambda\boldsymbol{S}_{k-1} + \left[\frac{1}{q(e_k)\tau_k} - \lambda\right]\frac{\boldsymbol{S}_{k-1}\boldsymbol{x}_k\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}}{\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k} \tag{6.12}$$

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + q(e_k)\boldsymbol{S}_k\boldsymbol{x}_k e_k \tag{6.13}$$

where $q(e_k)\boldsymbol{S}_k\boldsymbol{x}_k = \boldsymbol{S}_{k-1}\boldsymbol{x}_k/\tau_k$, $\tau_k = \boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k$, $\lambda > 1$ is the forgetting factor, and $q(e_k) = (1/e_k)\partial h(e_k)/\partial e_k$ is the gradient of the Hampel three-part redescending M-estimate function $h(e_k)$ [63, 64]. For an impulsive-noise corrupted e_k, $q(e_k)$ assumes a value of zero in which case we obtain $\boldsymbol{w}_k = \boldsymbol{w}_{k-1}$. However, $\boldsymbol{S}_k \neq \boldsymbol{S}_{k-1}$ although in [64] it is assumed that $\boldsymbol{S}_k = \boldsymbol{S}_{k-1}$. Applying the matrix-inversion lemma [3] to (6.12), we obtain the input-signal autocorrelation matrix as

$$\boldsymbol{R}_k = \frac{1}{\lambda}\left(\boldsymbol{R}_{k-1} - \frac{1}{\tau_k}\boldsymbol{x}_k\boldsymbol{x}_k^T\right) + q(e_k)\boldsymbol{x}_k\boldsymbol{x}_k^T \tag{6.14}$$

As can be seen, the update equation of the autocorrelation matrix forgets the present and past input signal vectors in which case its positive definiteness can be lost in low-precision implementations [56, 55]. Therefore, the RQN algorithm would exhibit explosive divergence with $\lambda \gg 1$; on the other hand, with $\lambda \approx 1$ it could lose its tracking capability. It is important to note that the absence of explosive divergence is the main feature of the QN algorithm as reported in [58].

For the proposed RQN algorithms, we obtain $\alpha_k \subset (0, 1)$ for the error bounds given in (6.5) in each adaptation and, therefore, the Theorems 1 and 2 in the chapter 5 can be readily applied to the proposed RQN algorithms. As a result, as per Theorem 3 of chapter 5, the proposed RQN algorithms are stable and asymptotically stable.

6.3 MSE Analysis of Proposed RQN Algorithms

In this section, we analyze the MSE behavior of the proposed RQN algorithms in stationary and nonstationary environments. It should be mentioned that as in the KQN and PQN algorithms, the excess MSE (EMSE) expressions in (5.45) and (5.47) which are actually derived for the LMS-Newton algorithms in [59] still hold true for the proposed RQN algorithms for the reasons discussed in section 5.4. However, the approach used in [59] for deriving the EMSE involves certain assumptions [59]. In this section, we use the framework of the energy conservation relation recently proposed in [20], [87] for analyzing steady-state performance of adaptive filters as it involves no assumptions. As a result, the energy conservation relation has become the most popular approach nowadays and has been effectively used to analyze several adaptation algorithms in [95], [96], [97].

6.3.1 Excess MSE in Stationary Environments

The weight-vector update equation in (6.2) can be expressed in terms of weight-error vector $\hat{\boldsymbol{w}}_k = \boldsymbol{w}_o - \boldsymbol{w}_k$ as

$$\hat{\boldsymbol{w}}_k = \hat{\boldsymbol{w}}_{k-1} - \frac{\alpha_k}{\tau_k}\boldsymbol{S}_{k-1}\boldsymbol{x}_k e_k \tag{6.15}$$

where $\boldsymbol{w}_o$ denotes the weight vector of the unknown system and $\tau_k = \boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k$. Premultiplying both sides of (6.15) by input-signal vector $\boldsymbol{x}_k$, we obtain

$$\boldsymbol{x}_k^T\hat{\boldsymbol{w}}_k = \boldsymbol{x}_k^T\hat{\boldsymbol{w}}_{k-1} - \alpha_k e_k \tag{6.16}$$

Now, we can define the noiseless *a priori* and noiseless *a posteriori* errors as

$$e_{f,k} = \boldsymbol{x}_k^T\hat{\boldsymbol{w}}_{k-1} \tag{6.17}$$

and

$$\epsilon_{f,k} = \boldsymbol{x}_k^T\hat{\boldsymbol{w}}_k \tag{6.18}$$

respectively. Using (6.17) and (6.18) in (6.16) yields

$$\epsilon_{f,k} = e_{f,k} - \alpha_k e_k \tag{6.19}$$

Now if we use (6.19) in (6.15), we obtain

$$\hat{\boldsymbol{w}}_k + \frac{e_{f,k}}{\tau_k}\boldsymbol{S}_{k-1}\boldsymbol{x}_k = \hat{\boldsymbol{w}}_{k-1} + \frac{\epsilon_{f,k}}{\tau_k}\boldsymbol{S}_{k-1}\boldsymbol{x}_k \tag{6.20}$$

If we take the square of the weighted L_2 norm on both sides of (6.20) and do some manipulation, we obtain

$$\|\hat{\boldsymbol{w}}_k\|^2_{\boldsymbol{R}_{k-1}} + \frac{e^2_{f,k}}{\tau_k} = \|\hat{\boldsymbol{w}}_{k-1}\|^2_{\boldsymbol{R}_{k-1}} + \frac{\epsilon^2_{f,k}}{\tau_k} \tag{6.21}$$

where $\boldsymbol{R}_k$ is a positive definite matrix which is obtained by taking of the inverse of $\boldsymbol{S}_k$ given by (5.15) as

$$\boldsymbol{R}_k = \boldsymbol{R}_{k-1} + \frac{\alpha_k}{(1-\alpha_k)\tau_k}\boldsymbol{x}_k\boldsymbol{x}_k^T \tag{6.22}$$

which according to Theorem 1 in section (5.2) would remain a positive define matrix indefinitely if initialized with a positive definite matrix. The energy conservation relation can now be obtained by taking the expectation on both sides of (6.21) as

$$E\left[\|\hat{\boldsymbol{w}}_k\|^2_{\boldsymbol{R}_{k-1}}\right] + E\left[\frac{e^2_{f,k}}{\tau_k}\right] = E\left[\|\hat{\boldsymbol{w}}_{k-1}\|^2_{\boldsymbol{R}_{k-1}}\right] + E\left[\frac{\epsilon^2_{f,k}}{\tau_k}\right] \tag{6.23}$$

As can be seen, no assumption has been made in deriving the energy conservation relation for the proposed RQN algorithms. Now at steady-state we obtain $E\left[\|\hat{\boldsymbol{w}}_k\|^2_{\boldsymbol{R}_{k-1}}\right] = E\left[\|\hat{\boldsymbol{w}}_{k-1}\|^2_{\boldsymbol{R}_{k-1}}\right]$ and, therefore, the energy conservation relation becomes

$$E\left[\frac{e^2_{f,k}}{\tau_k}\right] = E\left[\frac{\epsilon^2_{f,k}}{\tau_k}\right] \tag{6.24}$$

Using (6.19) in (6.24), we obtain

$$2E\left[\frac{\alpha_k e_{f,k} e_k}{\tau_k}\right] = E\left[\frac{\alpha^2_k e^2_k}{\tau_k}\right] \tag{6.25}$$

The noisy *a priori* error in (6.4) can be expressed in terms of the noiseless *a priori* error in (6.17) as

$$e_k = e_{f,k} + v_k \tag{6.26}$$

where v_k is the measurement noise. Therefore, (6.25) can be expressed as in terms of (6.26) as

$$2E\left[\frac{\alpha_k e^2_{f,k}}{\tau_k}\right] + 2E\left[\frac{\alpha_k e_{f,k} v_k}{\tau_k}\right] = E\left[\frac{\alpha^2_k e^2_{f,k}}{\tau_k}\right] + E\left[\frac{\alpha^2_k v^2_k}{\tau_k}\right] + 2E\left[\frac{\alpha^2_k e_{f,k} v_k}{\tau_k}\right] \tag{6.27}$$

Since in practice the measurement noise v_k is independent and identically distributed with respect to the input-desired signal pairs and has a zero mean, we obtain

$$2E\left[\frac{\alpha_k e^2_{f,k}}{\tau_k}\right] = E\left[\frac{\alpha^2_k e^2_{f,k}}{\tau_k}\right] + E\left[\frac{\alpha^2_k}{\tau_k}\right]\sigma^2_v \tag{6.28}$$

where σ^2_v is the variance of the measurement noise. Note that no assumption was made in reaching (6.28). However, in order to proceed further, we make the assumptions that (1) τ_k is independent of parameter α_k and (2) any dependency of τ_k on $e_{f,k}$ can be neglected, which are valid during steady-state of the adaptive filter. In addition, at steady state $\sigma_{1,k}/|e_k| \rightarrow 1$ and $\gamma = |e_k| - \nu\theta_k$ and hence for a small value of ν, (6.3) yields

$$\alpha_k \approx 1.98\nu \tag{6.29}$$

Thus (6.28) simplifies to

$$E\left[e_{f,k}^2\right] = \frac{\alpha_k}{2-\alpha_k}\sigma_v^2 \tag{6.30}$$

Therefore, the EMSE of the proposed RQN algorithms in stationary environment becomes

$$\text{EMSE} = \frac{\alpha_k}{2-\alpha_k}\sigma_v^2 \tag{6.31}$$

As can be seen, if we replace α_k by a constant p, known as reduction factor, as was done in section 5.4, we obtain (5.45).

6.3.2 Excess MSE in Nonstationary Environments

A nonstationary unknown system is modeled as

$$\boldsymbol{w}_{o,k} = \boldsymbol{w}_{o,k-1} + \boldsymbol{\phi}_k \tag{6.32}$$

where the optimal weight-vector $\boldsymbol{w}_{o,k}$ is represented by a first-order Markov model [101] and the elements of $\boldsymbol{\phi}_k$ are zero-mean white Gaussian noise samples with variance σ_ϕ^2. The weight-vector update equation in (6.2) for the system model in (6.32) can be expressed in terms of the weight error vector as

$$\hat{\boldsymbol{w}}_k = \hat{\boldsymbol{w}}_{k-1} - \frac{\alpha_k}{\tau_k}\boldsymbol{S}_{k-1}\boldsymbol{x}_k e_k - \boldsymbol{\phi}_k \tag{6.33}$$

where

$$\hat{\boldsymbol{w}}_k = \boldsymbol{w}_{o,k-1} - \boldsymbol{w}_k \tag{6.34}$$

and

$$\hat{\boldsymbol{w}}_{k-1} = \boldsymbol{w}_{o,k-1} - \boldsymbol{w}_{k-1} \tag{6.35}$$

Premultiplying both sides of (6.33) by $\boldsymbol{x}_k^T$, we obtain

$$\boldsymbol{x}_k^T\hat{\boldsymbol{w}}_k = \boldsymbol{x}_k^T\hat{\boldsymbol{w}}_{k-1} - \alpha_k e_k - \boldsymbol{x}_k^T\boldsymbol{\phi}_k \tag{6.36}$$

The relations in (6.17) and (6.18) hold true for the model in (6.32). Therefore, using (6.17) and (6.18) in (6.36), we obtain

$$\epsilon_{f,k} = e_{f,k} - \alpha_k e_k - \boldsymbol{x}_k^T\boldsymbol{\phi}_k \tag{6.37}$$

Now using (6.37) in (6.33), we obtain

$$\hat{\boldsymbol{w}}_k + \frac{e_{f,k}}{\tau_k}\boldsymbol{S}_{k-1}\boldsymbol{x}_k = \hat{\boldsymbol{w}}_{k-1} + \frac{\epsilon_{f,k}}{\tau_k}\boldsymbol{S}_{k-1}\boldsymbol{x}_k + \frac{\boldsymbol{x}_k^T\boldsymbol{\phi}_k}{\tau_k}\boldsymbol{S}_{k-1}\boldsymbol{x}_k - \boldsymbol{\phi}_k \tag{6.38}$$

If we take the weighted L_2 norm on both sides of (6.38), we obtain

$$\begin{aligned}\|\hat{\boldsymbol{w}}_k\|^2_{\boldsymbol{R}_{k-1}} + \frac{e^2_{f,k}}{\tau_k} = \|\hat{\boldsymbol{w}}_{k-1}\|^2_{\boldsymbol{R}_{k-1}} + \frac{\epsilon^2_{f,k}}{\tau_k} + 2\frac{e_{f,k}\boldsymbol{x}_k^T\boldsymbol{\phi}_k}{\tau_k} - 2\hat{\boldsymbol{w}}^T_{k-1}\boldsymbol{R}_{k-1}\boldsymbol{\phi}_k \\ + \|\boldsymbol{\phi}_k\|^2_{\boldsymbol{R}_{k-1}} - \frac{\boldsymbol{x}_k^T\boldsymbol{\phi}_k\boldsymbol{\phi}_k^T\boldsymbol{x}_k}{\tau_k}\end{aligned} \tag{6.39}$$

Taking the expectation on both sides, we obtain the energy conversation relation for the system (6.32) as

$$\begin{aligned}E\left[\|\hat{\boldsymbol{w}}_k\|^2_{\boldsymbol{R}_{k-1}}\right] + E\left[\frac{e^2_{f,k}}{\tau_k}\right] = E\left[\|\hat{\boldsymbol{w}}_{k-1}\|^2_{\boldsymbol{R}_{k-1}}\right] + E\left[\frac{\epsilon^2_{f,k}}{\tau_k}\right] + 2E\left[\frac{\boldsymbol{x}_k^T\hat{\boldsymbol{w}}_{k-1}\boldsymbol{\phi}_k^T\boldsymbol{x}_k}{\tau_k}\right] \\ -2E\left[\hat{\boldsymbol{w}}^T_{k-1}\boldsymbol{R}_{k-1}\boldsymbol{\phi}_k\right] + E\left[\|\boldsymbol{\phi}_k\|^2_{\boldsymbol{R}_{k-1}}\right] \\ -E\left[\frac{\boldsymbol{x}_k^T\boldsymbol{\phi}_k\boldsymbol{\phi}_k^T\boldsymbol{x}_k}{\tau_k}\right]\end{aligned} \tag{6.40}$$

Since vector $\boldsymbol{\phi}_k$ in (6.40) is independent of the weight-error vector $\hat{\boldsymbol{w}}_{k-1}$ and $\boldsymbol{x}_k$ and it is obtained from the samples of a zero-mean white Gaussian noise signal, the energy conversation relation in (6.40) simplifies to

$$\begin{aligned}E\left[\|\hat{\boldsymbol{w}}_k\|^2_{\boldsymbol{R}_{k-1}}\right] + E\left[\frac{e^2_{f,k}}{\tau_k}\right] = E\left[\|\hat{\boldsymbol{w}}_{k-1}\|^2_{\boldsymbol{R}_{k-1}}\right] + E\left[\frac{\epsilon^2_{f,k}}{\tau_k}\right] + E\left[\|\boldsymbol{\phi}_k\|^2_{\boldsymbol{R}_{k-1}}\right] \\ -E\left[\frac{\boldsymbol{x}_k^T\boldsymbol{\phi}_k\boldsymbol{\phi}_k^T\boldsymbol{x}_k}{\tau_k}\right]\end{aligned} \tag{6.41}$$

Note that no assumption has been made to obtain the simplified energy conversation relation in (6.41). During steady state, we obtain $E\left[\|\hat{\boldsymbol{w}}_k\|^2_{\boldsymbol{R}_{k-1}}\right] = E\left[\|\hat{\boldsymbol{w}}_{k-1}\|^2_{\boldsymbol{R}_{k-1}}\right]$ and, therefore, the energy conservation relation assumes the form

$$E\left[\frac{e^2_{f,k}}{\tau_k}\right] = E\left[\frac{\epsilon^2_{f,k}}{\tau_k}\right] + E\left[\|\boldsymbol{\phi}_k\|^2_{\boldsymbol{R}_{k-1}}\right] - E\left[\frac{\boldsymbol{x}_k^T\boldsymbol{\phi}_k\boldsymbol{\phi}_k^T\boldsymbol{x}_k}{\tau_k}\right] \tag{6.42}$$

Since $\boldsymbol{\phi}_k$ is an independent zero-mean white Gaussian noise signal, some straightforward simplifications can be made on the relation obtained by using $\epsilon_{f,k}$ given by

(6.37) in (6.42), which result in

$$2E\left[\frac{\alpha_k e_k e_{f,k}}{\tau_k}\right] = E\left[\frac{\alpha_k^2 e_k^2}{\tau_k}\right] + E\left[\|\boldsymbol{\phi}_k\|^2_{\boldsymbol{R}_{k-1}}\right] \tag{6.43}$$

Since the measurement noise in (6.26) is an independent white Gaussian noise signal, (6.26) and (6.43) give

$$2E\left[\frac{\alpha_k e_{f,k}^2}{\tau_k}\right] = E\left[\frac{\alpha_k^2 e_{f,k}^2}{\tau_k}\right] + E\left[\frac{\alpha_k^2}{\tau_k}\right]\sigma_v^2 + E\left[\|\boldsymbol{\phi}_k\|^2_{\boldsymbol{R}_{k-1}}\right] \tag{6.44}$$

Invoking assumptions (1) and (2) in section 6.3.1 and (6.29), we obtain

$$E\left[e_{f,k}^2\right] = \frac{\alpha_k}{2-\alpha_k}\sigma_v^2 + \frac{E[\tau_k]E\left[\|\boldsymbol{\phi}_k\|^2_{\boldsymbol{R}_{k-1}}\right]}{2\alpha_k - \alpha_k^2} \tag{6.45}$$

The values of $E[\tau_k]$ and $E\left[\|\boldsymbol{\phi}_k\|^2_{\boldsymbol{R}_{k-1}}\right]$ can be obtained as

$$E[\tau_k] = E[\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k] = \mathrm{tr}\{\boldsymbol{S}_{k-1}E[\boldsymbol{x}_k\boldsymbol{x}_k^T]\} = \mathrm{tr}\{\boldsymbol{S}_{k-1}\boldsymbol{R}\} \approx M \tag{6.46}$$

and

$$E\left[\|\boldsymbol{\phi}_k\|^2_{\boldsymbol{R}_{k-1}}\right] = E\left[\boldsymbol{\phi}_k^T\boldsymbol{R}_{k-1}\boldsymbol{\phi}_k\right] = \mathrm{tr}\{\boldsymbol{R}_{k-1}E\left[\boldsymbol{\phi}_k\boldsymbol{\phi}_k^T\right]\} \approx M\sigma_x^2\sigma_\phi^2 \tag{6.47}$$

Using (6.46) and (6.47) in (6.45), we obtain

$$\mathrm{EMSE} = \frac{\alpha_k}{2-\alpha_k}\sigma_v^2 + \frac{M^2\sigma_x^2\sigma_\phi^2}{2\alpha_k - \alpha_k^2} \tag{6.48}$$

As can be seen, if we replace α_k by constant p as in section 5.4, we obtain (5.47).

6.4 Practical Considerations

In low-cost fixed-point hardware implementations, the accumulation of roundoff errors can cause $\boldsymbol{S}_k$ to lose its positive definiteness. This problem can be eliminated by periodically reinitializing $\boldsymbol{S}_k$ using the identity matrix. This is also done in the QN algorithm in [58] whenever $\boldsymbol{x}_k^T\boldsymbol{S}_{k-1}\boldsymbol{x}_k < 10^{-3}$ for the case of a fixed-point implementation and also in the classical optimization context. However, very frequent reinitializations could slow the convergence of the algorithm. A compromise number of reinitializations can be achieved by reinitializing $\boldsymbol{S}_k$ whenever $|e_k| > 2.576\sigma_k$ in which case the probability of reinitialization in a given iteration assumes the value of 0.01. Reinitialization of other parameters is not required.

The variance estimator of the error signal in (6.7) should be initialized with a large value to ensure that the probability that $\gamma = \gamma_c$ in the transient state is increased. With $\gamma = \gamma_c$, the transient state of the adaptive filter would die out quickly. Although a rough choice of σ_0^2 would work, we have used $\sigma_0^2 = c_1 M/\sigma_v^2$ where $0 < c_1 < 1$ is a positive constant. This choice yields good results. The forgetting factor in (6.7) and (6.9) are chosen as $\lambda = 1 - 1/(c_2 M)$ and $\beta = 1 - 1/(c_3 M)$, respectively, where $c_2 > (1/M)$ and $c_3 > (1/M)$ are positive scalars [68]. The length P of vector $\boldsymbol{g}_k$ should be sufficiently greater than the duration of the impulsive noise. A reduced ν would yield a reduced steady-state misalignment and improved robustness with respect to impulsive noise. On the other hand, increased values of γ_c and θ_k would reduce the number of updates and yield a reduced steady-state misalignment. However, the convergence speed would be compromised in such a situation.

6.5 Simulation Results

In order to evaluate the performance of the proposed RQN with fixed γ_c (PRQN-I) and the proposed RQN with variable γ_c (PRQN-II) algorithms, a system identification application was considered as detailed below. For the sake of comparison, simulations were carried out with the known QN (KQN) and the known RQN (KRQN) algorithms described in [58] and [64], respectively.

In the first experiment, the adaptive filter was used to identify a 36th-order lowpass FIR filter with a normalized cutoff frequency of 0.3. No additive white Gaussian noise was added to the coefficients of the FIR filter. The input signal was generated by filtering a white Gaussian noise signal with zero mean and unity variance through an FIR filter with a single pole at 0.95. The eigenvalue spread ratio of the colored input signal of 10000 samples was obtained by using the ensemble average of the autocorrelation matrix given by (1.28) over 1000 independent trials with $\lambda = 1 - 2^{-15}$ and $\boldsymbol{R}_0 = 10^{-4}\boldsymbol{I}$ as 839. The measurement noise added to the desired signal was a white Gaussian noise signal with zero mean and variances of 10^{-3} and 10^{-6} to achieve SNRs of 30 and 60 dB, respectively. The impulse response of the FIR filter was suddenly multiplied by -1 at iteration 2000 to check the tracking capability of the algorithm. Impulsive noise of duration T_s where T_s is the sampling duration was added to the desired signal at iterations 1000, 1300, and 3300 using a Bernoulli trial with probability 0.001 [102]. The updates in the KRQN algorithm were carried out only when $q(e_k) \neq 0$. We have explored two options in the implementation of the KRQN algorithm, first using

$$\boldsymbol{S}_k = \lambda \boldsymbol{S}_{k-1} + \left\{ \left[\frac{1 - \lambda q(e_k)\tau_k}{q(e_k)\tau_k^2} \right] \boldsymbol{S}_{k-1}\boldsymbol{x}_k \right\} \boldsymbol{x}_k^T \boldsymbol{S}_{k-1} \tag{6.49}$$

and then using (6.12) for the evaluation of $\boldsymbol{S}_k$. Although (6.12) and (6.49) are equivalent, it turns out that the implementation of (6.49) is subject to numerical ill-conditioning, which can cause instability as demonstrated in the second experiment. On the other hand, the implementation of (6.12) was found to be more robust although it requires increased computational effort. We have used (6.12) in our implementation of the KRON algorithm. The learning curves obtained from 1000 independent trials by using the KQN, KRQN, PRQN-I and PRQN-II algorithms are illustrated in Figs. 6.1 to 6.2. As can be seen, algorithms PRQN-I and PRQN-II offer similar performance; both are robust with respect to impulsive noise and yield significantly reduced steady-state misalignment. As may be expected, the KQN algorithm is seriously compromised by impulsive noise both in terms of robustness and tracking capability. The total number of updates required by the PRQN-I and PRQN-II algorithms were 1333 and 1414 for SNR of 30 dB, and 1353 and 1271, for SNR of 60 dB, respectively, as compared to 4000 in the other algorithms. Note that two systems were identified in this experiment, one before iteration 2000 and the other after iteration 2000. Otherwise, the number of updates would have been reduced by half.

The second experiment was identical to the first experiment except that the elements of the weight vector of the FIR filter were contaminated by an additive zero-mean white Gaussian noise signal as per (6.32). The algorithm parameters were set to identical values in all algorithms as in the first experiment except that ν was set to 0.3 in the PRQN-I and PRQN-II algorithms. We have used (6.49) in our implementation of the KRON algorithm. The learning curves obtained in 1000 independent trials are illustrated in Figs. 6.3–6.4. As can be seen, the PRQN-I and PRQN-II algorithms offer robust performance with respect to impulsive noise, reduced steady-state misalignment, and fast tracking compared to the other algorithms. In the KQN algorithm, the convergence speed is compromised when impulsive noise is encountered

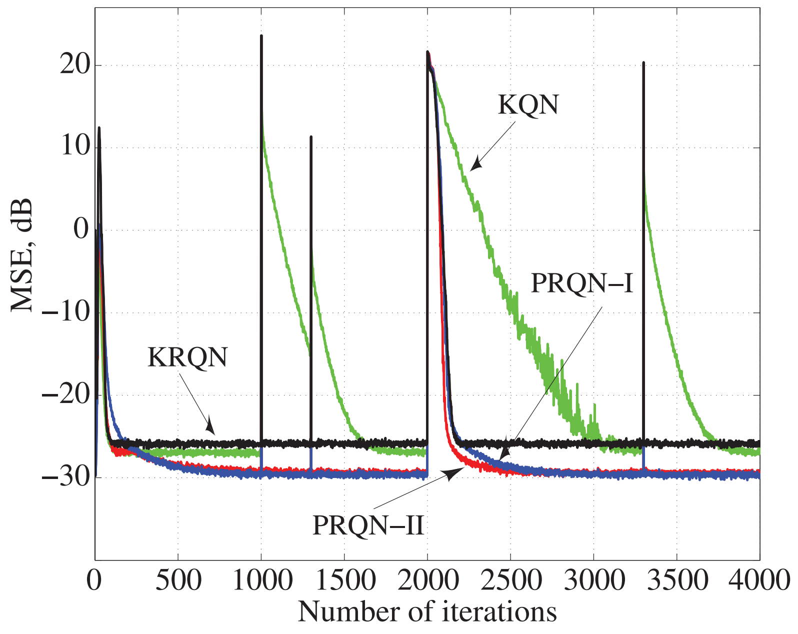

Figure 6.1: Learning curves for proposed and competing algorithms. $\boldsymbol{S}_0 = \boldsymbol{I}$ and $\boldsymbol{w}_0 = \boldsymbol{0}$ in all algorithms. Parameters for the PRQN-I algorithm: $\nu = 0.5$, $P = 1$, $c_1 = 1$, $c_2 = 4$. Parameters for the PRQN-II algorithm: $\nu = 0.5$, $P = 15$, $\xi_0 = 10$, $\hat{\sigma}_0^2 = 10$, $c_3 = 2$. Parameters for the KRQN algorithm were set as suggested in [64].

as in the first experiment. The KRQN algorithm, on the other hand, like the RLS algorithm, exhibits explosive divergence [8,9,10] and reduced re-adaptation capability. The number of weight-vector updates in the PRQN-I and PRQN-II algorithms were 1016 and 2238 for SNR of 30 dB, and 921 and 2008 for SNR of 60 dB, respectively, as compared to 4000 in the other algorithms.

In the third and fourth experiments, we verified the accuracy of the expression of EMSE in Eqs. (6.31) and (6.48), respectively. In these two experiments, the order of the FIR filter was set to 24 and its coefficients were not multiplied by -1 at iteration 2000 as we only wanted to check the accuracy of (6.31) and (6.48). The algorithm parameters in these two experiments were set to the same values as in the first experiment.

In the third experiment, we used the relation $\text{MSE} = \sigma_v^2 + \text{EMSE}$ where EMSE is given in (6.31) to obtain the theoretical steady-state MSEs of the proposed algorithms as given in Table 6.1. As can be seen, the theoretical values match reasonably well the experimental values.

In the fourth experiment, the elements of the weight vector of the FIR filter

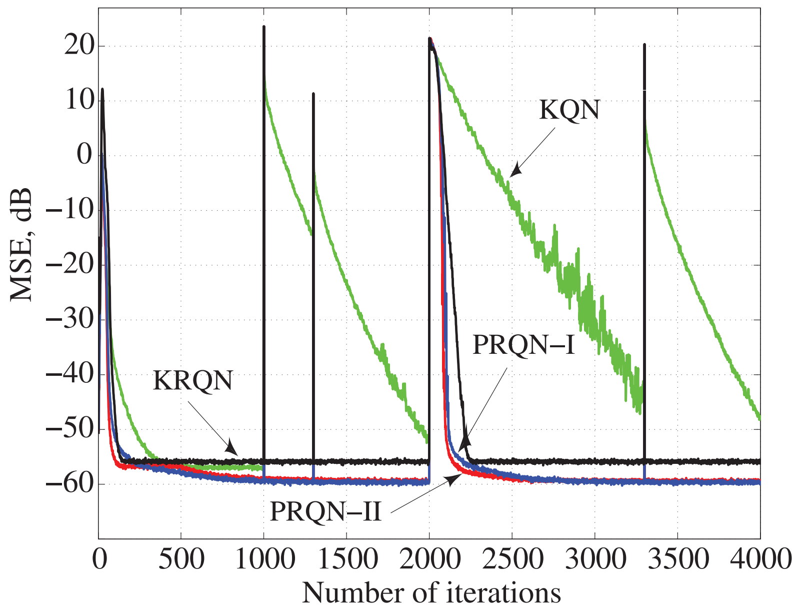

Figure 6.2: Learning curves for proposed and competing algorithms. $\boldsymbol{S}_0 = \boldsymbol{I}$, and $\boldsymbol{w}_0 = \boldsymbol{0}$ in all algorithms. Parameters for the PRQN-I algorithm: $\nu = 0.5$, $P = 1$, $c_1 = 1$, $c_2 = 4$. Parameters for the PRQN-II algorithm: $\nu = 0.5$, $P = 15$, $\xi_0 = 10$, $\hat{\sigma}_0^2 = 10$, $c_3 = 2$. Parameters for the KRQN algorithm were set as suggested in [64].

were contaminated by a zero-mean white Gaussian noise signal as per (6.32). The theoretical steady-state MSEs of the proposed algorithms were obtained by using (6.48) in the relation MSE $= \sigma_v^2 +$ EMSE and are given in Table. 6.2. As can be seen, the theoretical values match reasonably well the experimental values.

6.6 Conclusions

Two new robust quasi-Newton adaptation algorithms have been developed on the basis of the mathematical framework of the basic classical QN optimization algorithm, which lead to an improved estimate of the inverse of the Hessian. Like the data-selective QN algorithm we described in chapter 5, the proposed RQN algorithms incorporate data selective adaptation which significantly reduces the number of adaptations required. An MSE analysis for the proposed RQN algorithms was presented. Like the PQN algorithm described in chapter 5, the proposed RQN algorithms are also asymptotically stable. Simulation results obtained in the case of a system identification application demonstrate that the proposed RQN algorithms converge much

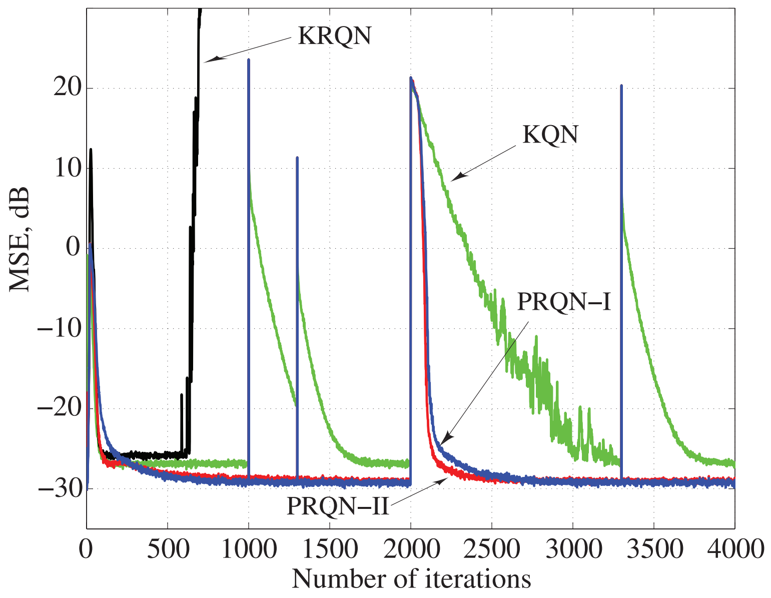

Figure 6.3: Learning curves for proposed and competing algorithms. The variance of the additive white Gaussian noise signal was $\sigma_\phi^2 = 10^{-9}$.

faster than the known QN algorithm reported in [58] for medium and high SNRs. On the other hand, the known QN is not robust against impulsive noise and the known RQN algorithm reported in [64] has reduced tracking capability. Simulation results obtained also demonstrate that the theoretical steady-state MSE of the proposed RQN algorithms matches reasonably well the experimental steady-state MSE.

In the next chapter, we propose an adaptation algorithm based on the minimum-error entropy criterion.

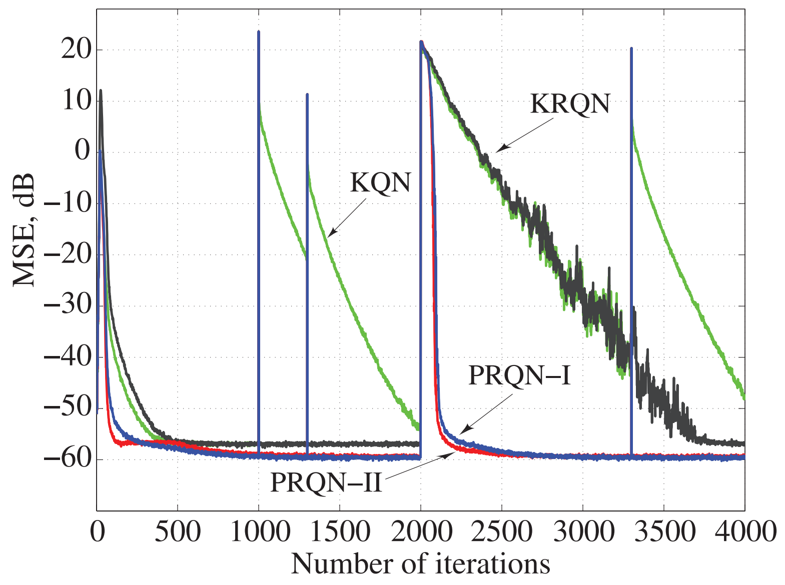

Figure 6.4: Learning curves for proposed and competing algorithms. Parameter λ for the KRQN algorithm was set to $\lambda = 1/0.99999$. The variance of the additive white Gaussian noise signal was $\sigma_\phi^2 = 10^{-13}$.

Table 6.1: Steady-State MSE in dB in Proposed RQN Algorithms
$M = 24,\ \sigma_v^2 = 10^{-3}$

ν	PRQN-I	PRQN-II	Eq. (6.31)
0.05	-29.7	-29.6	-29.8
0.10	-29.8	-29.7	-29.5
0.15	-29.8	-29.7	-29.3
0.20	-29.7	-29.6	-29.0
0.25	-29.7	-29.6	-28.8
0.30	-29.6	-29.5	-28.5
0.35	-29.4	-29.3	-28.2
0.40	-29.4	-29.3	-27.8
0.45	-29.2	-29.1	-27.7
0.50	-29.2	-29.1	-27.0

Table 6.2: Steady-State MSE in dB in Proposed RQN Algorithms
$M = 24,\ \sigma_v^2 = 10^{-4},\ \sigma_\phi^2 = 10^{-12}$

ν	PRQN-I	PRQN-II	Eq. (6.50)
0.05	-39.7	-39.6	-39.8
0.10	-39.8	-39.7	-39.6
0.15	-39.8	-39.6	-39.3
0.20	-39.6	-39.6	-39.0
0.25	-39.6	-39.5	-38.8
0.30	-39.5	-39.4	-38.5
0.35	-39.5	-39.4	-38.2
0.40	-39.3	-39.2	-37.8
0.45	-39.2	-39.1	-37.4
0.50	-39.2	-39.0	-37.0

Chapter 7

A New Normalized Minimum-Error Entropy Algorithm

7.1 Introduction

The algorithms described so far use the mean-square error (MSE) criterion which is suitable for applications where signals can be modeled in terms of Gaussian distributions. In other applications, improved performance can be achieved by using algorithms that are based on the minimum error-entropy (MEE) criterion such as those described in [74, 75].

In this chapter, we propose a new normalized minimum error-entropy (NMEE) algorithm [88] that is computationally simple and yields faster convergence and lower misadjustment than the stochastic MEE algorithm and the self-adjusting step size MEE (VMEE) algorithms reported in [76] and [77], respectively. The proposed NMEE algorithm is computationally simpler than the conventional NMEE algorithm reported in [78]. As in the conventional NMEE algorithm, the performance of the proposed NMEE algorithm is not influenced by the kernel size and the power of the input signal and, therefore, yields better performance as compared to the MEE and the VMEE algorithms.

7.2 Proposed NMEE Algorithm

An adaptive filter using Renyi's entropy [79] as an optimization criterion is illustrated in Fig. 7.1 where $\boldsymbol{x}_k$, d_k, and e_k are the input, desired, and error signals, respectively. The *a priori* error signal at k-th iteration is given by

$$e_k = d_k - \boldsymbol{x}_k^T \boldsymbol{w}_{k-1} \tag{7.1}$$

where $\boldsymbol{w}_{k-1}$ is the weight-vector at iteration $k-1$. The proposed NMEE algorithm

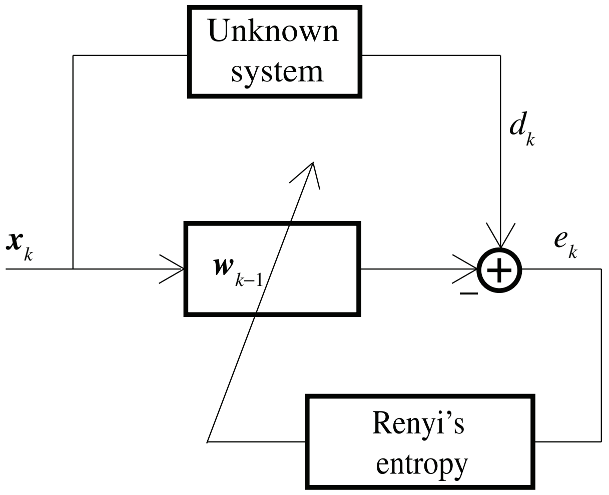

Figure 7.1: MEE adaptive-filter configuration.

is derived by solving the optimization problem

$$\underset{\boldsymbol{w}}{\text{minimize}} \;\; \|\boldsymbol{w}_k - \boldsymbol{w}_{k-1}\|^2 \tag{7.2}$$

subject to the constraint

$$V(0) - V(e_{p,k}) = 0$$

where $\boldsymbol{w}$ is the weight vector,

$$V(e_{p,k}) = \frac{1}{L} \sum_{i=k-L}^{k-1} \kappa_\sigma(e_{p,k} - e_{p,i}) \tag{7.3}$$

is the stochastic *a posteriori* information potential (IP) function (1.40), $V(0) = 1/\sigma\sqrt{2\pi}$ is the upper bound of $V(e)$, and $e_{p,i} = d_i - \boldsymbol{w}_k^T \boldsymbol{x}_i$ is the *a posteriori* error signal for $k - L \leq i \leq k$. By using the Lagrange multiplier method, the above optimization problem can be solved by minimizing the objective function

$$J_{\boldsymbol{w}_k} = ||\boldsymbol{w}_k - \boldsymbol{w}_{k-1}||^2 + \lambda[V(0) - V(e_{p,k})] \tag{7.4}$$

The gradient of (7.4) with respect to $\boldsymbol{w}_k$ becomes

$$\nabla J_{\boldsymbol{w}_k} = 2[\boldsymbol{w}_k - \boldsymbol{w}_{k-1}] - \lambda \frac{\partial V(e_{p,k})}{\partial \boldsymbol{w}_k} \tag{7.5}$$

Setting the gradient to zero, we get

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \frac{\lambda}{2} \frac{\partial V(e_{p,k})}{\partial \boldsymbol{w}_k} \tag{7.6}$$

The partial derivative of (7.3) with respect to $\boldsymbol{w}_k$ becomes

$$\nabla V(e_{p,k}) = c \sum_{i=k-L}^{k-1} \nu_{k,i} \boldsymbol{x}_{k,i} \tag{7.7}$$

where $c = 1/(2L\sigma^2)$, $\nu_{k,i} = (e_{p,k} - e_{p,i})\kappa_\sigma(e_{p,k} - e_{p,i})$ and $\boldsymbol{x}_{k,i} = \boldsymbol{x}_k - \boldsymbol{x}_i$. From (7.6) and (7.7), we get

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \lambda \frac{c}{2} \sum_{i=k-L}^{k-1} \nu_{k,i} \boldsymbol{x}_{k,i} \tag{7.8}$$

The *a posteriori* error $e_{p,i}$ for $k - L \leq i \leq k$ corresponding to the updated weight vector in (7.8) can be obtained as

$$\begin{aligned} e_{p,i} &= d_i - \boldsymbol{w}_{k-1}^T \boldsymbol{x}_i - \lambda \frac{c}{2} \left(\sum_{i=k-L}^{k-1} \nu_{k,i} \boldsymbol{x}_{k,i}^T \right) \boldsymbol{x}_i \\ &= e_{a,i} - \frac{\lambda}{2} \nabla V(e_{p,k})^T \boldsymbol{x}_i \quad \text{for } k - L \leq i \leq k \end{aligned} \tag{7.9}$$

where $e_i = d_i - \boldsymbol{w}_{k-1}^T \boldsymbol{x}_i$ for $k - L \leq i \leq k$ are the *a priori* errors. From (7.3) and (7.9), we obtain the *a posteriori* IP function as

$$\begin{aligned} V(e_{p,k}) &= \frac{1}{L} \sum_{i=k-L}^{k-1} \kappa_\sigma \left[e_{a,k} - \frac{\lambda}{2} \nabla V(e_{p,k})^T \boldsymbol{x}_{k,i} - e_{a,i} \right] \\ &= \frac{1}{L} \sum_{i=k-L}^{k-1} \kappa_\sigma(u_{i,k}) \end{aligned} \tag{7.10}$$

where

$$u_{i,k} = e_{a,k} - e_{a,i} - (\lambda/2) \nabla V(e_{p,k})^T \boldsymbol{x}_{k,i}$$

for $i = k - L, \ldots, k - 1$. The *a posteriori* IP function in (7.10) is equal to $V(0)$ if $\sum_{i=k-L}^{i=k-1} u_{i,k}^2 = 0$ as the Gaussian kernel is always positive. As can be seen, the solution of $\sum_{i=k-L}^{i=k-1} u_{i,k}^2 = 0$ yields two λ which can be imaginary. Therefore, we use the λ that minimizes $\sum_{i=k-L}^{i=k-1} u_{i,k}^2$ in (7.10), which is obtained by setting the derivative of $\sum_{i=k-L}^{i=k-1} u_{i,k}^2$ with respect to λ to zero i.e., $\sum_{i=k-L}^{i=k-1} \left[\nabla V(e_{p,k})^T \boldsymbol{x}_{k,i} \right] u_{i,k} = 0$. For a small L and small kernel size, $\nabla V(e_{p,k})^T \boldsymbol{x}_{k,i}$ can be assumed to be uniform and in such a case the Lagrange multiplier can be obtained by solving $\sum_{i=k-L}^{i=k-1} u_{i,k} = 0$ as [88]

$$\lambda = \frac{2}{\sum_{i=k-L}^{k-1} \left(\nabla V(e_{p,k})^T \boldsymbol{x}_{k,i} \right)} \sum_{i=k-L}^{k-1} (e_{a,k} - e_{a,i}) \tag{7.11}$$

Now from (7.11) and (7.6), the recursion formula assumes the form

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \frac{\nabla V_k(e_{p,k}) \sum_{i=k-L}^{k-1} (e_{a,k} - e_{a,i})}{\nabla V(e_{p,k})^T \sum_{i=k-L}^{k-1} \boldsymbol{x}_{k,i}} \tag{7.12}$$

As we can see in (7.12), the input data and the corresponding error signal e_i from iterations $i = k - L$ to $k - 1$ are reused at every iteration k and, therefore, the recursion formula in (7.12) gives better performance in the case of a colored input signal [88]. From (7.7) and (7.12), we obtain

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \frac{\sum_{i=k-L}^{k-1} \nu_{k,i} \boldsymbol{x}_{k,i} \sum_{i=k-L}^{k-1} (e_{a,k} - e_{a,i})}{\left(\sum_{i=k-L}^{k-1} \nu_{k,i} \boldsymbol{x}_{k,i} \right)^T \left(\sum_{i=k-L}^{k-1} \boldsymbol{x}_{k,i} \right)} \tag{7.13}$$

For a sufficiently small kernel size and window length, the above updating formula can be simplified further because the scaling factor $\nu_{k,i}$ can be assumed to be uniform over the Parzen window. Therefore, it can be factored out from the numerator and denominator in (7.13) to obtain

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \frac{\bar{\boldsymbol{x}}_{k,i}}{\bar{\boldsymbol{x}}_{k,i}^T \bar{\boldsymbol{x}}_{k,i}} \bar{e}_{a,k,i} \tag{7.14}$$

where $\bar{\boldsymbol{x}}_{k,i} = \sum_{i=k-L}^{k-1} \boldsymbol{x}_{k,i}$, and $\bar{e}_{a,k,i} = \sum_{i=k-L}^{k-1} (e_{a,k} - e_{a,i})$. The recursion formula in (7.14) significantly reduces the computational complexity of the algorithm.

In order to control the steady-state misalignment in the NMEE algorithm, a step size μ can be introduced in (7.14). However, the fastest convergence rate can be obtained with $\mu = 1$. The proposed NMEE algorithm will converge for values of μ in the range $0 < \mu < 2$.

7.3 Simulation Results

In order to compare the performance of the proposed NMEE (PNMEE) algorithm with that of the MEE, VMEE, and NLMS algorithms, we applied these algorithms in a plant identification problem and a chaotic time-series prediction problem. We have considered a plant with an impulse response given by

$$\boldsymbol{h}^T = [0.1\ 0.2\ 0.3\ 0.4\ 0.5\ 0.4\ 0.3\ 0.2\ 0.1]$$

In the first experiment, the input signal was Gaussian noise with unit power ($1p$). The step sizes of the MEE and the VMEE algorithms were chosen to be as large as possible to achieve the fastest convergence without degrading the robustness of the algorithms. The weight-error power curves for the MEE, VMEE, NLMS, and the PNMEE algorithms are plotted in Fig. 7.2. As can be seen, the PNMEE algorithm leads to faster convergence compared to the MEE, VMEE, and NLMS algorithms.

In the second experiment, we have used a Gaussian input signal with the power level increased by a factor of ten (10p) and the step sizes of the MEE and the VMEE algorithms were selected to be 5 times smaller than the values in the first experiment to assure the robustness of the algorithm. The weight-error power curves for the MEE, VMEE, NLMS, and the PNMEE algorithms are illustrated in Fig. 7.3. As can be seen, changes in the power of the input signal cause the convergence speed of the MEE and the VMEE algorithms to change significantly whereas that of the NLMS and the PNMEE algorithm do not change very much. The residual error

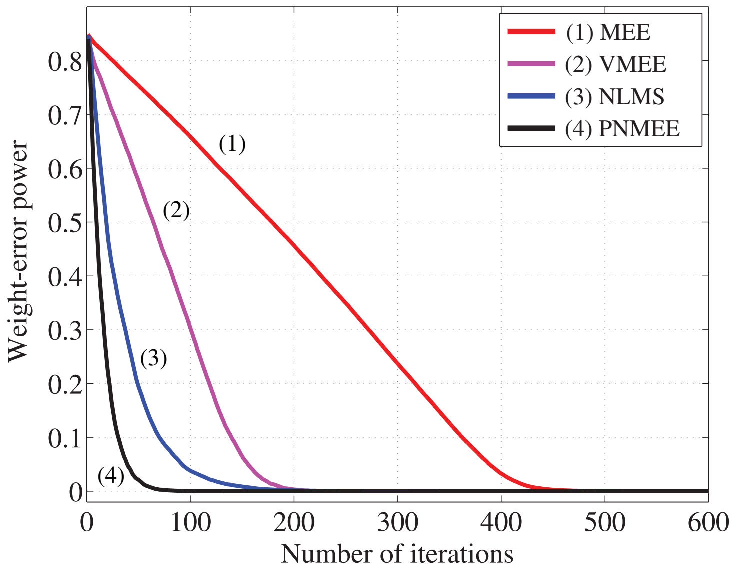

Figure 7.2: Convergence of weight-error power.

after 2000 iterations and the required number of iterations to converge for the various algorithms are given in Table 7.1. As can be seen, the PNMEE algorithm yields the lowest final residual error and the fastest convergence as compared to those of the other algorithms.

Table 7.1: Comparison of MEE Algorithms for White Input

	MEE	VMEE	NLMS	PNMEE
Res.	1.115×10^{-25}	1.888×10^{-29}	2.221×10^{-14}	3.741×10^{-32}
Iter., $1p$	450	200	200	90
Res.	1.944×10^{-7}	2.675×10^{-31}	2.211×10^{-28}	3.833×10^{-32}
Iter., $10p$	1550	950	280	100

In the third experiment, we repeated the first experiment except that the input signal was colored by a system with poles at $(-0.3, -0.7)$. The weight-error power curves obtained by using the MEE, VMEE, NLMS, and the PNMEE algorithms are illustrated in Fig. 7.4. As can be seen, the PNMEE algorithm requires fewer iterations than the other algorithms to converge. Note also that small fluctuations in the weight-error power are present at steady state in the case of the MEE algorithm. The final misadjustment of the algorithms are shown in Table 7.2. As can be seen,

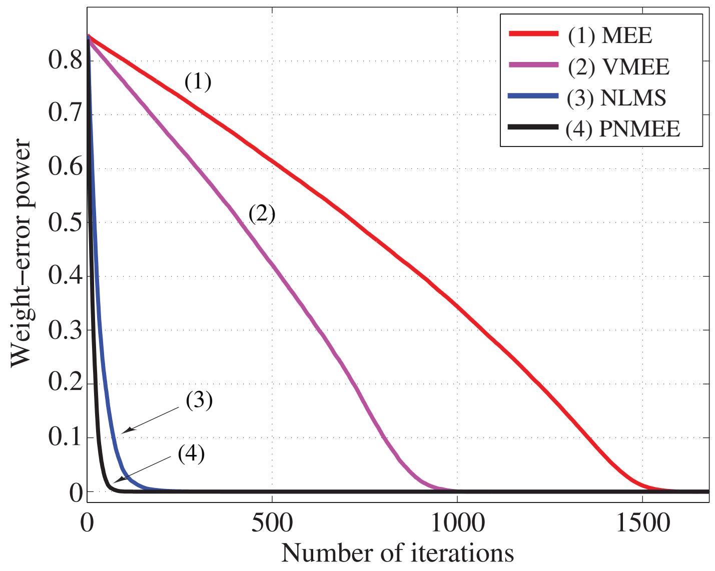

Figure 7.3: Convergence of weight-error power.

Table 7.2: Final Misadjustments for Colored Input, $1p$

	MEE	VMEE	NLMS	PNMEE
Res.	2.343×10^{-8}	1.389×10^{-11}	1.033×10^{-7}	3.474×10^{-32}

the PNMEE algorithm yields the lowest final residual error.

In the fourth experiment, we repeated the second experiment except that the input signal was colored by a system with poles at $(-0.3, -0.7)$. The weight-error power curves for the MEE, VMEE and the PNMEE algorithms for different kernel sizes are plotted in Fig. 7.5. As can be seen, the convergence speed of the algorithms increases with an increase in the kernel size which corroborates the observation in [103]. The misadjustments after 3000 iterations are given in Table 7.3 which shows

Table 7.3: Final Misadjustments for Colored Input, $10p$

kernel size	MEE	VMEE	PNMEE
0.4	1.061×10^{-3}	1.129×10^{-3}	2.960×10^{-32}
0.7	4.937×10^{-4}	1.720×10^{-5}	3.192×10^{-32}

that the PNMEE algorithm yields the lowest residual error.

In the fifth experiment, we used an FIR filter of length 6 to predict a Mackey-

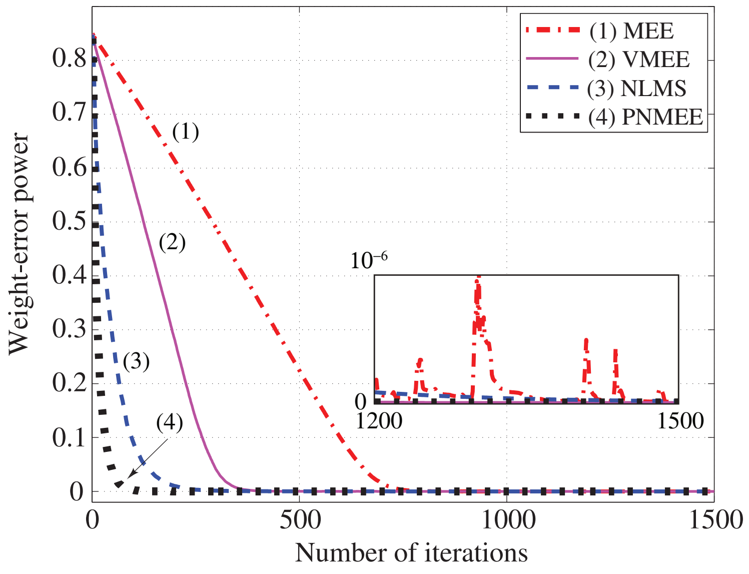

Figure 7.4: Convergence of weight-error power.

Glass time series [77, 89] with delay parameter $\tau = 30$. The window length and the kernel size were set to 50 and 0.01, respectively. The mean-squared error curves for the MEE, VMEE, and the PNMEE algorithms are shown in Fig. 7.6. As can be seen, the PNMEE algorithm offers faster convergence as compared to the MEE and the VMEE algorithms.

7.4 Conclusions

A new NMEE algorithm with a significantly reduced computational complexity has been proposed. The new algorithm offers faster convergence, lower misadjustment, and better tracking than the MEE and the VMEE algorithms. Furthermore, its performance does not depend on the power of the input signal. With data reusing, the proposed algorithm yields significantly lower final residual error and faster convergence than those of the MEE, NLMS, and VMEE algorithms for the case of a colored input signal.

In the next chapter, we describe a family of iterative-shrinkage adaptation algorithms.

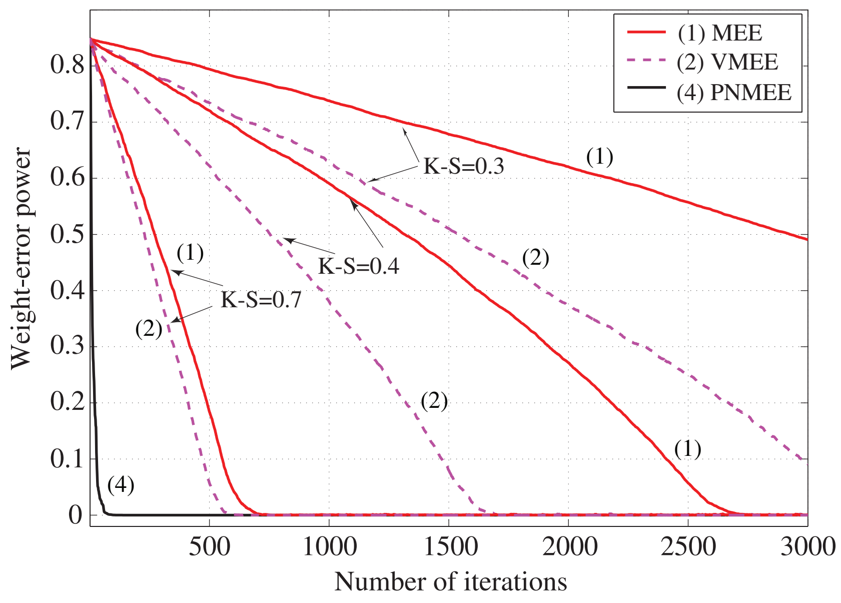

Figure 7.5: Convergence of weight-error power.

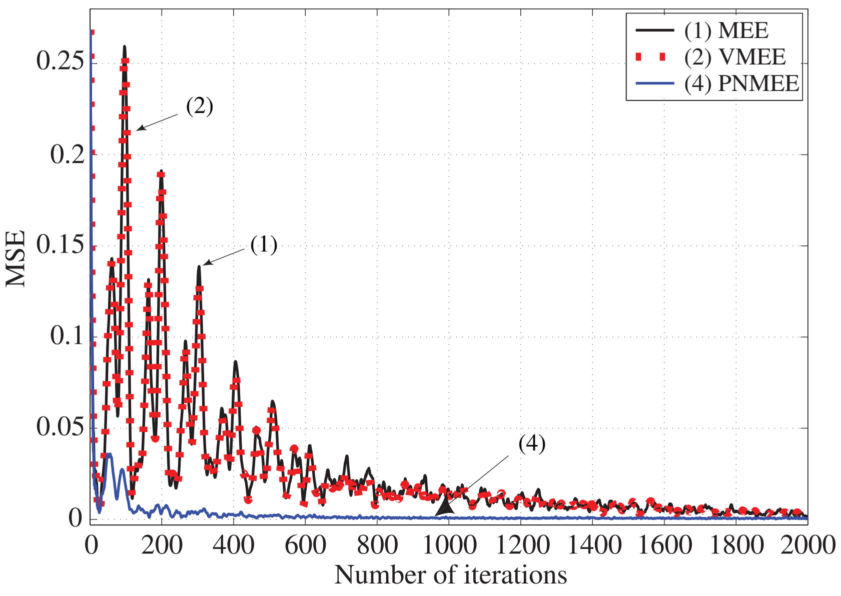

Figure 7.6: Mean squared error curve of MEE criterion

Chapter 8

A Family of Shrinkage Adaptation Algorithms

8.1 Introduction

In affine-projection (AP) algorithms (see section 1.1.3), information about the variance of the measurement noise is required to be available *a priori* for the successful operation of the set-membership AP (SMAP) algorithm, variable step size AP (VSSAP) algorithm, variable regularization AP algorithm, and the variable reuse time AP algorithm reported in [19], [21], [22], and [24], respectively. The algorithms in [19, 21] use a variable step size and the algorithm in [22] uses a variable regularization parameter both of which are obtained by using the noisy error signal. Due to the presence of noise in the error signal, the variable parameters in [19, 21, 22] can fail to reach the desired values in some applications. Hence, improved performance can be achieved if a noise-free error signal is used to obtain a variable step size in the conventional AP algorithm [11].

In this chapter, we propose a family of shrinkage adaptation algorithms, namely, the shrinkage AP (SHAP), shrinkage NLMS (SHNLMS), and shrinkage LMS (SHLMS) algorithms [90]. We apply the *iterative-shrinkage/threshold* methods described in [80, 81] to obtain a variable step size in the conventional AP algorithm, which is actually a solution of the $L_1 - L_2$ minimization problem used in signal denoising applications. We have used the iterative-shrinkage/threshold method to obtain a noise-free *a priori* error signal whose average power is used to obtain the variable step size. In this way, the variable step size can reach a lower value during steady state leading to a reduced steady-state misalignment. The variable step size actually minimizes the energy of the noise-free *a posteriori* error signal. The SHAP algorithm yields a significantly reduced steady-state misalignment while preserving the fast convergence of the AP algorithm. Unlike the SMAP and VSSAP algorithms in [19] and [21], respectively, the SHAP algorithm uses the noise-free error signal to obtain a variable step size that reaches a much lower value during steady state. The same is true for the SHNLMS and SHLMS algorithms. Simulation results in a system identification application and an acoustic-echo cancelation application are used to demonstrate the superior performance of the proposed algorithms as compared to that of the variable step size LMS (VLMS) algorithm, non-parametric normalized LMS (NPNLMS) algorithm, conventional AP algorithm, SMAP algorithm, VSSAP algorithm, and the SMNLMS algorithm reported in [6], [10], [11], [19], [21], and [34].

8.2 Shrinkage Affine-Projection Algorithm

In the case of a system identification adaptive-filtering application, the desired signal samples d_k are obtained as

$$d_k = \boldsymbol{x}_k^T \boldsymbol{w}_{opt} + v_k \tag{8.1}$$

where $\boldsymbol{w}_{opt} \in \mathcal{R}^{M\times 1}$ is the impulse response of the unknown system, $\boldsymbol{x}_k \in \mathcal{R}^{M\times 1}$ is the input-signal vector, and v_k is the measurement noise signal. The conventional AP algorithm uses the weight-vector update equation

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \mu \boldsymbol{X}_k (\boldsymbol{X}_k^T \boldsymbol{X}_k)^{-1} \boldsymbol{e}_k \tag{8.2}$$

where μ is the step size, $\boldsymbol{X}_k \in \mathcal{R}^{M\times L}$ is the input-signal matrix which is obtained as $\boldsymbol{X}_k = [\boldsymbol{x}_k \ \boldsymbol{x}_{k-1} \ \cdots \ \boldsymbol{x}_{k-L+1}]$, and $\boldsymbol{e}_k \in \mathcal{R}^{L\times 1}$ is the *a priori* error-signal vector. Parameter L is known as the projection order of the AP algorithm. The *a priori* error vector $\boldsymbol{e}_k$ for the AP algorithm is obtained as

$$\boldsymbol{e}_k = \boldsymbol{e}_{f,k} + \boldsymbol{v}_k \tag{8.3}$$

where

$$\boldsymbol{e}_{f,k} = \boldsymbol{X}_k^T \left(\boldsymbol{w}_{opt} - \boldsymbol{w}_{k-1}\right) \tag{8.4}$$

is the noise-free *a priori* error vector and $\boldsymbol{v}_k = [v_k \; v_{k-1} \; \cdots \; v_{k-L+1}]^T$ contains the measurement noise signal samples. Similarly, the *a posteriori* error vector can be expressed as

$$\boldsymbol{\epsilon}_k = \boldsymbol{\epsilon}_{f,k} + \boldsymbol{v}_k \tag{8.5}$$

where

$$\boldsymbol{\epsilon}_{f,k} = \boldsymbol{X}_k^T \left(\boldsymbol{w}_{opt} - \boldsymbol{w}_k\right) \tag{8.6}$$

is the noise-free *a posteriori* error vector. The update equation in (8.2) yields the noise-free *a posteriori* error vector as

$$\boldsymbol{\epsilon}_{f,k} = (1-\mu)\boldsymbol{e}_{f,k} - \mu\boldsymbol{v}_k \tag{8.7}$$

Taking the expectation of the square of the L_2 norm of $\boldsymbol{\epsilon}_{f,k}$ in (8.7), we obtain

$$E\left[\|\boldsymbol{\epsilon}_{f,k}\|^2\right] = (1-\mu)^2 E\left[\|\boldsymbol{e}_{f,k}\|^2\right] + \mu^2 E\left[\|\boldsymbol{v}_k\|^2\right] \tag{8.8}$$

To achieve $\boldsymbol{w}_k = \boldsymbol{w}_{opt}$ at steady state, we require that $E\left[\|\boldsymbol{\epsilon}_{f,k}\|^2\right] = 0$. Unfortunately, as can be seen from (8.8), in such a situation μ would become imaginary. In other words, there is no real μ that would force the equality $E\left[\|\boldsymbol{\epsilon}_{f,k}\|^2\right] = 0$. However, we can minimize $E\left[\|\boldsymbol{\epsilon}_{f,k}\|^2\right]$ with respect to μ and the optimal value of μ can be obtained by setting the gradient of $E\left[\|\boldsymbol{\epsilon}_{f,k}\|^2\right]$ with respect to μ to zero. In this way, the optimal value of μ can be obtained as

$$\mu_{opt} = \frac{E\left[\|\boldsymbol{e}_{f,k}\|^2\right]}{E\left[\|\boldsymbol{e}_{f,k}\|^2\right] + E\left[\|\boldsymbol{v}_k\|^2\right]} \tag{8.9}$$

As can be seen, the step size μ_{opt} would lie in the range $(0, 1)$. Since $E\left[\|e_{f,k}\|^2\right]$ is a measure of the excess mean-square error (EMSE) of the conventional AP algorithm [20], using (8.9) in (8.2) the minimum EMSE can be achieved.

One obvious difficulty in computing the step size in (8.9) is to obtain $E\left[\|\boldsymbol{e}_{f,k}\|^2\right]$. Although the time average of the squares of $\boldsymbol{e}_{f,k}$ i.e.,

$$\sigma_{e,f,k}^2 = \lambda\sigma_{e,f,k-1}^2 + (1-\lambda)\|\boldsymbol{e}_{f,k}\|^2 \tag{8.10}$$

where $0 \ll \lambda < 1$ is the forgetting factor, can be used to replace the statistical mean $E\left[\|\boldsymbol{e}_{f,k}\|^2\right]$ in (8.9), the problem cannot be solved as $\boldsymbol{e}_{f,k}$ is unknown. An easy solution would be to recover $\boldsymbol{e}_{f,k}$ from the noisy *a priori* error vector $\boldsymbol{e}_k$ in (8.3). Once we recover $\boldsymbol{e}_{f,k}$ from $\boldsymbol{e}_k$, we can use it in (8.10) to obtain an estimate of $E\left[\|\boldsymbol{e}_{f,k}\|^2\right]$. On the other hand, we can obtain $E\left[\|\boldsymbol{v}_k\|^2\right] = L\sigma_v^2$, where L is the projection order. With both $E\left[\|\boldsymbol{e}_{f,k}\|^2\right]$ and $E\left[\|\boldsymbol{v}_k\|^2\right]$ known, the step size in (8.9) can be easily computed. Since the norm of $\boldsymbol{e}_{f,k}$ is used in (8.10), μ_{opt} cannot approach zero unless the adaptive filter reaches steady state.

8.2.1 Shrinkage Denoising Method

In this subsection, we discuss the recovery of $\boldsymbol{e}_{f,k}$ from the noisy *a priori* error vector $\boldsymbol{e}_k$. We consider the minimization problem

$$\underset{\boldsymbol{a}_k}{\operatorname{argmin}} \left(t\|\boldsymbol{a}_k\|_1 + 0.5\|\boldsymbol{D}\boldsymbol{a}_k - \boldsymbol{e}_k\|^2\right) \tag{8.11}$$

where $\boldsymbol{D} \in \mathcal{R}^{L\times L}$ is an orthonormal matrix and t is a threshold parameter. This type of optimization problem is used to solve image denoising problems [80, 81]. The solution of the minimization problem in (8.11) can be obtained in a straightforward way as

$$\boldsymbol{a}_k = \operatorname{sign}(\boldsymbol{a}_{o,k}) \odot \max\left(|\boldsymbol{a}_{o,k}| - t, 0\right) \tag{8.12}$$

where $\boldsymbol{a}_{o,k} = \boldsymbol{D}^T\boldsymbol{e}_k$ [80, 81] and $\odot$ denotes the element-wise product. With an appropriate threshold t, the minimization problem in (8.11) essentially aims at recovering $\boldsymbol{e}_{f,k}$ from $\boldsymbol{e}_k$ in (8.3). With $\boldsymbol{a}_k$ in (8.12) known, we obtain an estimate of $\boldsymbol{e}_{f,k}$ in (8.4) as $\hat{\boldsymbol{e}}_{f,k} = \boldsymbol{D}\boldsymbol{a}_k$. The estimate $\hat{\boldsymbol{e}}_{f,k}$ can now be used in (8.10) to obtain an estimate of $E\left[\|\boldsymbol{e}_{f,k}\|^2\right]$ to be used in (8.9). In the next subsection, we determine the threshold parameter t.

8.2.2 Choice of Threshold Parameter

The elements of $\boldsymbol{v}_k$ can be chosen to be samples of a zero-mean white Gaussian noise signal with variance σ_v^2 and, therefore, from (8.3) we obtain

$$E\left[\|\boldsymbol{e}_k\|^2\right] = E\left[\|\boldsymbol{e}_{f,k}\|^2\right] + L\sigma_v^2 \tag{8.13}$$

As can be seen from (8.13), the threshold parameter should be chosen as $t = \sqrt{L\sigma_v^2}$. With such a choice, (8.12) would shrink all of the elements of $|\boldsymbol{a}_k|$ towards zero by the amount t. Each shrinkage operation would bring significant reduction in the noise component $\boldsymbol{v}_k$ from the *a priori* error vector $\boldsymbol{e}_k$ in (8.3). As a result, we would obtain

$\hat{\boldsymbol{e}}_{f,k} = \boldsymbol{D}\boldsymbol{a}_k \approx \boldsymbol{e}_{f,k}$. Therefore, the update equations for the shrinkage AP algorithm become [90]

$$\boldsymbol{a}_{o,k} = \boldsymbol{D}^T\boldsymbol{e}_k \tag{8.14}$$
$$\boldsymbol{a}_k = \text{sign}(\boldsymbol{a}_{o,k}) \odot \max\left(|\boldsymbol{a}_{o,k}| - t, 0\right) \tag{8.15}$$
$$\hat{\boldsymbol{e}}_{f,k} = \boldsymbol{D}\boldsymbol{a}_k \tag{8.16}$$
$$\sigma^2_{e,f,k} = \lambda\sigma^2_{e,f,k-1} + (1-\lambda)\|\hat{\boldsymbol{e}}_{f,k}\|^2 \tag{8.17}$$
$$\mu_k = \frac{\sigma^2_{e,f,k}}{\sigma^2_{e,f,k} + L\sigma^2_v} \tag{8.18}$$
$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \mu_k\boldsymbol{X}_k(\boldsymbol{X}_k^T\boldsymbol{X}_k)^{-1}\boldsymbol{e}_k \tag{8.19}$$

A different orthonormal matrix $\boldsymbol{D}$ would yield different performance as will be shown in the simulation results. Since the identity matrix is also an orthonormal matrix, using $\boldsymbol{D} = \boldsymbol{I}$ and $L = 1$ we obtain the shrinkage NLMS algorithm.

It should be mentioned that $E\left[\|\boldsymbol{e}_{f,k}\|^2\right]$ can be obtained directly from (8.13) as $E\left[\|\boldsymbol{e}_{f,k}\|^2\right] = E\left[\|\boldsymbol{e}_k\|^2\right] - L\sigma^2_v$ and it can then be used in (8.9) to obtain μ. However, since we have to use the time average of $\|\boldsymbol{e}_k\|^2$ instead of its statistical average, i.e., $E\left[\|\boldsymbol{e}_k\|^2\right]$, in (8.9) and the relation in (8.13) does not hold true for the time average, the algorithm would lose its performance in such situations and can become unstable as μ can assume values outside the range $(1, 0)$.

The energy of the *a posteriori* error signal in (8.5) for the shrinkage AP algorithm becomes

$$E[\|\boldsymbol{\epsilon}_k\|^2] = \frac{(E[\|\boldsymbol{v}_k\|^2])^2}{E\left[\|\boldsymbol{e}_{f,k}\|^2\right] + E\left[\|\boldsymbol{v}_k\|^2\right]} \tag{8.20}$$

which is less than $E[\|\boldsymbol{v}_k\|^2]$. Note that $\boldsymbol{\epsilon}_{f,k}$ in (8.6) is not independent of $\boldsymbol{v}_k$ as can be seen from (8.1)–(8.3). As a result, a relation such as that of (8.13) cannot be obtained by using (8.5). Hence, we can obtain $E[\|\boldsymbol{\epsilon}_k\|^2] < E[\|\boldsymbol{v}_k\|^2]$ even though $E\left[\|\boldsymbol{\epsilon}_{f,k}\|^2\right]$ is a positive quantity for all L. Using $L = 1$ in (8.14)–(8.19), we obtain the SHNLMS algorithm. For the NPNLMS algorithm in [10], the energy of the *a posteriori* error signal becomes $E[\epsilon_k^2] = pE[v_k^2] + (1-p)E[e_k^2] > E[v_k^2]$ where p is the probability of update. In adaptive filter theory, $E\left[\|e_{f,k}\|^2\right] \neq 0$ for $\mu \neq 0$ and, therefore, from (8.20) we obtain $E[\epsilon_k^2] < E[v_k^2]$ for the SHNLMS algorithm. A similar statement applies to the SHAP algorithm when compared with the AP algorithm designed for speech signals in [23]. In the next section, we describe the shrinkage LMS algorithm.

8.3 Shrinkage LMS Algorithm

The update formula for the conventional LMS algorithm is

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \mu e_k \boldsymbol{x}_k \tag{8.21}$$

where μ is the step size. The noise-free *a posteriori* error signal for the conventional LMS algorithm becomes

$$\epsilon_{f,k} = (1 - \mu \boldsymbol{x}_k^T \boldsymbol{x}_k) e_{f,k} - \mu \boldsymbol{x}_k^T \boldsymbol{x}_k v_k \tag{8.22}$$

Taking the expectation of the square of the noise-free *a posteriori* error signal in (8.22), we obtain

$$E[\epsilon_{f,k}^2] = E[(1 - \mu \boldsymbol{x}_k^T \boldsymbol{x}_k)^2 e_{f,k}^2] + \mu^2 E[(\boldsymbol{x}_k^T \boldsymbol{x}_k)^2] E[v_k^2] \tag{8.23}$$

where we assume that v_k is independent and identically distributed Gaussian noise signal with zero mean. In order to proceed further, we neglect the dependency of $e_{f,k}$ on the input signal $\boldsymbol{x}_k$ as the amplitudes of $e_{f,k}$ become quite small during steady state. Equation (8.23) can now be simplified as

$$\begin{aligned} E[\epsilon_{f,k}^2] = {} & E[1 - 2\mu \boldsymbol{x}_k^T \boldsymbol{x}_k + \mu^2 (\boldsymbol{x}_k^T \boldsymbol{x}_k)^2] E[e_{f,k}^2] \\ & + \mu^2 E[(\boldsymbol{x}_k^T \boldsymbol{x}_k)^2] E[v_k^2] \end{aligned} \tag{8.24}$$

Setting the derivative of $E[\epsilon_{f,k}^2]$ with respect to μ to zero, we obtain

$$\mu = \frac{1}{E[\|\boldsymbol{x}_k\|^2]} \, \frac{E[e_{f,k}^2]}{E[e_{f,k}^2] + \sigma_v^2} \tag{8.25}$$

where we assume $E[(\boldsymbol{x}_k^T \boldsymbol{x}_k)^2] = E[\boldsymbol{x}_k^T \boldsymbol{x}_k]^2$ in order to obtain a simple expression for μ. Since the *a priori* error signal e_k is a scalar quantity, $e_{f,k}$ can be recovered from e_k by using the shrinkage method in a trivial way as $\hat{e}_{f,k} = \text{sign}(e_k)\max(|e_k| - t, 0)$ where t is the threshold parameter. Following the discussion in section 8.2.2, we see that the threshold parameter t should be chosen as $t = \sigma_v$ where σ_v^2 is the variance of the noise signal. However, since the dependency of $e_{f,k}$ on the input signal $\boldsymbol{x}_k$ was neglected in deriving (8.25), the step size in (8.25) does not assure the minimization of $E[\epsilon_{f,k}^2]$. As a result, fine tuning of the threshold parameter t around σ_v could yield improved performance. From our extensive simulation results we found out that $t = \sqrt{Q\sigma_v^2}$ with $Q = 1$ to 4 works well. Based on these principles, the shrinkage LMS algorithm becomes

$$\hat{e}_{f,k} = \text{sign}(e_k) \odot \max\left(|e_k| - t, 0\right) \tag{8.26}$$

$$\sigma_{e,f,k}^2 = \lambda\sigma_{e,f,k-1}^2 + (1-\lambda)\hat{e}_{f,k}^2 \tag{8.27}$$

$$\mu_k = \frac{q}{E[\|\boldsymbol{x}_k\|^2]}\,\frac{\sigma_{e,f,k}^2}{\sigma_{e,f,k}^2 + \sigma_v^2} \tag{8.28}$$

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \mu_k\boldsymbol{x}_k e_k \tag{8.29}$$

where we have used the time average in place of the statistical average $E[e_{f,k}^2]$ in (8.25) and $q \in (0,1)$ as discussed in the next section. In practice, we assume that $E[\|\boldsymbol{x}_k\|^2]$ is known *a priori* as we can preset the input signal statistics in many practical applications. For a zero-mean white Gaussian input signal, $E[\|\boldsymbol{x}_k\|^2]$ can easily be obtained as $M\sigma_x^2$ where σ_x^2 is the variance of the input signal. It should be mentioned that information about $E[\|\boldsymbol{x}_k\|^2]$ is required to be known for stable operation for the conventional LMS algorithm as discussed below.

8.3.1 Stability of Shrinkage LMS Algorithm

The update formula for the LMS algorithm in (8.21) can be expressed in terms of weight-error vector $\hat{\boldsymbol{w}}_k = \boldsymbol{w}_{opt} - \boldsymbol{w}_k$ as

$$\hat{\boldsymbol{w}}_k = \left(\boldsymbol{I} - \mu\boldsymbol{x}_k\boldsymbol{x}_k^T\right)\hat{\boldsymbol{w}}_{k-1} - \mu_k\boldsymbol{x}_k v_k \tag{8.30}$$

Taking the expectation on both sides of (8.30), we obtain the evolution of $\hat{\boldsymbol{w}}_k$ as

$$E[\hat{\boldsymbol{w}}_k] = [\boldsymbol{I} - \mu\boldsymbol{R}]\,E[\hat{\boldsymbol{w}}_{k-1}] \tag{8.31}$$

where $\boldsymbol{R} = E[\boldsymbol{x}_k\boldsymbol{x}_k^T]$ is the correlation matrix. The evolution of the mean weight-error vector in (8.31) is governed by the modes of matrix $\boldsymbol{G} = [\boldsymbol{I} - \mu\boldsymbol{R}]$. Thus a necessary and sufficient condition for stability of the evolution of $E[\hat{\boldsymbol{w}}_k]$ of the LMS algorithm is that the matrix $\boldsymbol{G}$ be stable, i.e., all of its eigenvalues are required to be inside the unit circle [3]. This is equivalent to the condition

$$0 < \mu < \frac{2}{\lambda_{max}(\boldsymbol{R})} \tag{8.32}$$

where $\lambda_{max}(\boldsymbol{R})$ is the maximum eigenvalue of $\boldsymbol{R}$ [3]. For stability of $E[\|\hat{\boldsymbol{w}}_k\|^2]$, the step size of the LMS algorithm should satisfy the condition [3]

$$0 < \mu < \frac{2}{2\lambda_{max}(\boldsymbol{R}) + \sum_{i=1}^{M} \lambda_i} \tag{8.33}$$

where λ_i is the ith eigenvalue of $\boldsymbol{R}$. However, in practice the step size should not be chosen close to the upper bound in (8.33) as the steps in computing μ involve certain approximations and assumptions [3]. If M is assumed to be large and $E[x_k] = 0$, we have $E[\|\boldsymbol{x}_k\|^2] \approx \text{tr}\{E[\boldsymbol{x}_k\boldsymbol{x}_k^T]\} = \text{tr}\{\boldsymbol{R}\} = \sum_{i=1}^{M} \lambda_i$. Therefore, the step size in (8.28) for the proposed algorithm lies in the range

$$0 < \mu_k < \frac{q}{\sum_{i=1}^{M} \lambda_i} \tag{8.34}$$

with $q \in (0, 1)$ which clearly satisfies the stability condition of the LMS algorithm given in (8.32) and (8.33). Therefore, the step size in (8.28) for the proposed shrinkage LMS algorithm assures stability in terms of the evolution of $E[\hat{\boldsymbol{w}}_k]$ and $E[\|\hat{\boldsymbol{w}}_k\|^2]$.

8.4 Discussion

A difficulty associated with the implementation of shrinkage algorithms has to do with the availability of information about the noise variance.

Some recent applications of adaptive-filtering, which include adaptive mobile networks, source localization, environment monitoring, etc, have emerged recently [104, 105, 106, 107, 108], which require a set of sensors distributed over a region and each sensor requires an adaptive filter. Two topologies, namely, fusion and network topologies, are used for distributed computing. In the fusion topology, the central sensor can be used to obtain the noise variance and then transmit the information to all the other sensors. In the network topology, each sensor can be used to estimate the noise variance during the communication interval with the neighboring sensors. In acoustic-echo cancelation applications, on the other hand, the noise variance can be estimated during off periods in the speech signal.

Since the proposed shrinkage algorithms are actually variants of the basic LMS, NLMS, and AP algorithms, they are very reliable. The increased computational complexity associated with (8.14), (8.16), and (8.17) for the SHAP algorithm is of order $\mathcal{O}(L^2)$ which remains at a marginal level for a small value of L. Therefore, the proposed algorithms can be applied in many recent engineering applications of adaptive filters.

8.5 Simulation Results

In this section, the performance of the proposed algorithms is demonstrated in two adaptive filtering applications. The proposed SHAP algorithm is compared with the NLMS, AP, SMAP [19], and VSSAP [21] algorithms. The proposed SHNLMS algorithm is compared with the NLMS, NPNLMS [10], and SMNLMS [34] algorithms. The proposed SHLMS algorithm is compared with the conventional LMS and VLMS [6] algorithms. Regularization matrix $\delta \boldsymbol{I}$ with $\delta = 10^{-8}$ was added in all experiments and in all AP type algorithms to matrix $\boldsymbol{X}_k^T \boldsymbol{X}_k$ in (8.2) to assure its invertibility. The initial weight vector $\boldsymbol{w}_0$ was set to the zero vector in all algorithms and in all experiments. The error bound for the SM algorithms was set to $\gamma = \sqrt{5\sigma_v^2}$ [34, 19] in all experiments. Unless otherwise stated, the orthonormal matrix $\boldsymbol{D}$, forgetting factor λ, and $\sigma_{e,f,0}$ for the SHAP algorithm were chosen as the discrete cosine transform matrix, 0.90, and 0, respectively, in all experiments. The learning curves were obtained by averaging the mean-square error (MSE) or the mean-square deviation (MSD), defined as $MSD = 20 \log_{10} (\|\boldsymbol{w}_{opt} - \boldsymbol{w}_k\| / \|\boldsymbol{w}_{opt}\|)$, over 1000 trials in each experiment.

8.5.1 System-Identification Application

A series of experiments were carried out in a system-identification application where the unknown system was an FIR filter which was obtained as $\boldsymbol{w}_{opt} = \text{fir1}(M-1, 0.3)$ using MATLAB, where M is the filter order. The elements of the weight vector $\boldsymbol{w}_{opt}$ were normalized so that $\boldsymbol{w}_{opt}$ has unit norm.

In the first experiment, the order of the unknown system was set to 27 and the input signal was a zero-mean white Gaussian noise signal with unity variance. The input signal was colored by an IIR filter with transfer function [19]

$$H(z) = \frac{1}{z^4 - 0.95z^3 - 0.19z^2 - 0.09z + 0.5} \tag{8.35}$$

and the measurement noise added to the desired signal was also a white Gaussian noise signal with zero mean and variance of $\sigma_v^2 = 10^{-2}$. The eigenvalue spread ratio of the colored input signal of 10000 samples was obtained as 1531 by using the ensemble average of the autocorrelation matrix given by (1.28) over 1000 independent trials with $\lambda = 1 - 2^{-15}$ and $\boldsymbol{R}_0 = 10^{-4}\boldsymbol{I}$. The projection order L was set to 2 in all AP-type algorithms. The learning curves obtained are illustrated in Fig. 8.1. As

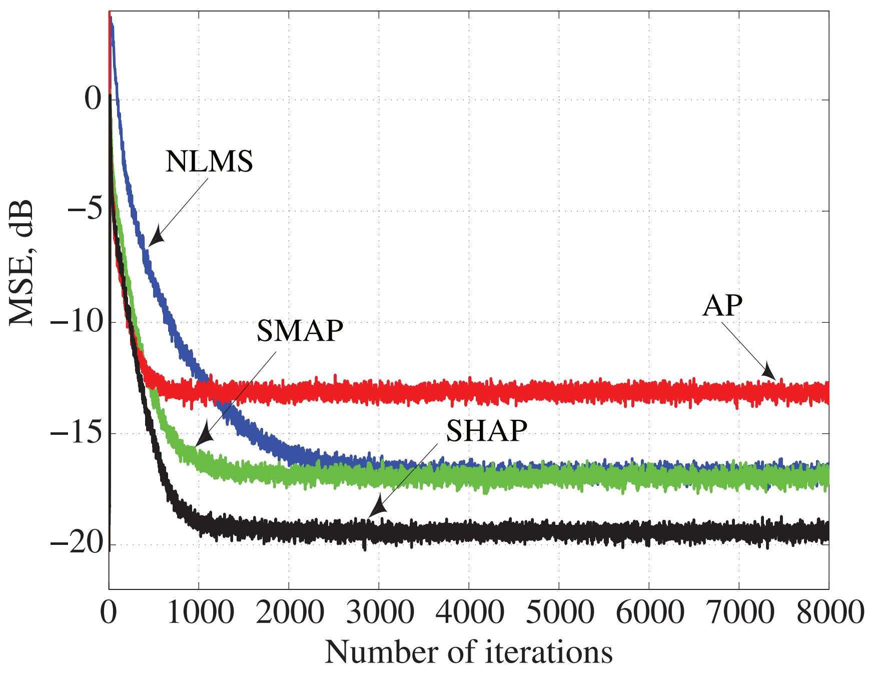

Figure 8.1: Learning curves with $L = 2$, $t = \sqrt{L\sigma_v^2}$.

can be seen, the proposed SHAP algorithm yields a significantly reduced steady-state misalignment for the same convergence speed as compare to the other algorithms.

In the second experiment, the order of the unknown system and projection order were changed to 37 and 4, respectively. In addition, the variance of the measurement noise was changed to 10^{-4}. The eigenvalue spread ratio of the colored input signal of 10000 samples was obtained as 1974 by using the ensemble average of the autocorrelation matrix given by (1.28) over 1000 independent trials with $\lambda = 1 - 2^{-15}$ and $\boldsymbol{R}_0 = 10^{-4}\boldsymbol{I}$. The learning curves obtained are illustrated in Fig. 8.2. As can be seen, the proposed SHAP algorithm yields a significantly reduced steady-state misalignment as compared to the other algorithms. The NLMS algorithm yields a reduced stead-state misalignment as compared to the AP and SMAP algorithm due to their increased projection order.

In the third experiment, we reduced the degree of the correlation of the input

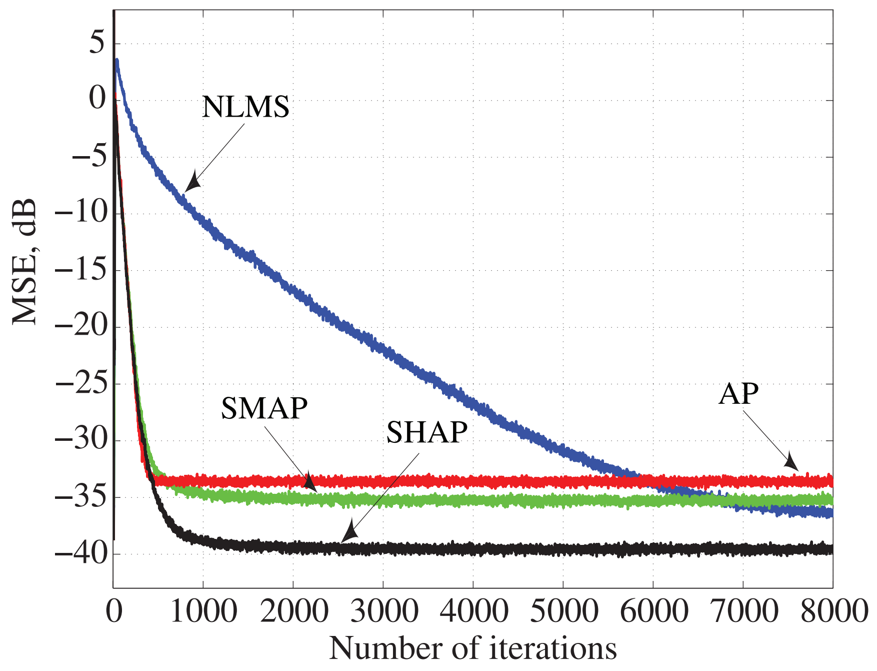

Figure 8.2: Learning curves with $L = 4$, $t = \sqrt{L\sigma_v^2}$.

signal of the adaptive filter by using an IIR filter with a single pole at 0.95 to filter the zero-mean unity-variance Gaussian signal. The eigenvalue spread ratio of the colored input signal of 10000 samples was obtained as 1276 by using the ensemble average of the autocorrelation matrix given by (1.28) over 1000 independent trials with $\lambda = 1 - 2^{-15}$ and $\boldsymbol{R}_0 = 10^{-4}\boldsymbol{I}$. The order of the unknown system was set to 111 and the variance of the measurement noise was set to 10^{-3}. The projection order was set to $L = 6$ in all algorithms. The learning curves obtained are illustrated in Fig. 8.3. As can be seen, the SHAP algorithm yields much reduced steady-state misalignment as compared to the other AP algorithms for the same convergence speed.

In the fourth experiment, we have repeated the third experiment with $M = 63$, $L = 3$, and the variance of the measurement noise was set to 0.0316. The eigenvalue spread ratio of the colored input signal of 10000 samples was obtained as 1081 by using the ensemble average of the autocorrelation matrix given by (1.28) over 1000 independent trials with $\lambda = 1 - 2^{-15}$ and $\boldsymbol{R}_0 = 10^{-4}\boldsymbol{I}$. The learning curves obtained are illustrated in Fig. 8.4. As can be seen, the SHAP algorithm yields a reduced steady-state misalignment as compared to the other AP algorithms for the same convergence speed.

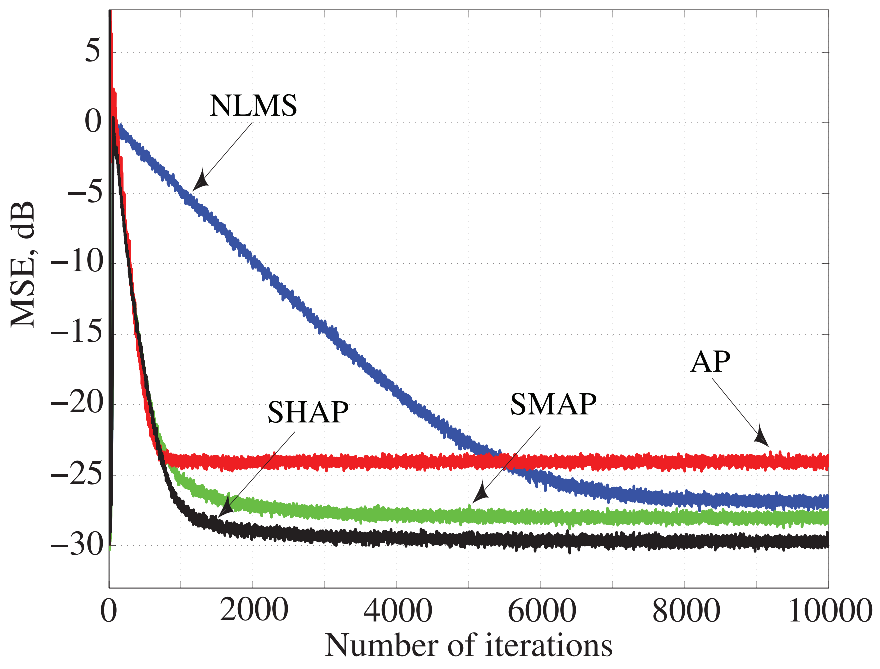

Figure 8.3: Learning curves with $L = 6$, $t = \sqrt{L\sigma_v^2}$.

In the fifth experiment, we investigated the effect of using different orthonormal matrices $\boldsymbol{D}$ on the convergence characteristics of the SHAP algorithm. We repeated the first experiment with $L = 8$ and $\lambda = 0.93$ using the AP algorithm and the SHAP algorithm with $\boldsymbol{D}$ as the identity matrix (SHAP-I), DCT matrix (SHAP-DCT), and Daubechies wavelet matrix (SHAP-WT). The MSD curves are illustrated in Fig. 8.5. As can be seen, the SHAP-DCT, and SHAP-WT algorithms perfrom better than the AP and SHAP-I algorithms. Since the amplitude of the error signal during steady state becomes very small, different $\boldsymbol{D}$'s would yield similar reduction in the elements of $\boldsymbol{e}_k$ and, as a result, similar noise removal. Consequently, different $\boldsymbol{D}$'s would result in similar values of μ_k in (8.18) and similar steady-state misalignment during steady state.

In the sixth experiment, we examined the performance of the SHAP algorithm with $L = 1$ (SHNLMS) and that of the NLMS, SMNLMS [34], and NPNLMS [10] algorithms. The order of the unknown system was set to 63. The input signal was obtained by filtering a zero-mean unity-variance white Gaussian noise signal by an IIR filter with a pole at 0.9. The variance of the measurement noise was set to 10^{-3}. The parameters for the NPNLMS algorithm were set to $\lambda = 1-1/(6M)$, $\delta = 10^{-8}$, $\hat{\sigma}_e^2 = 0$.

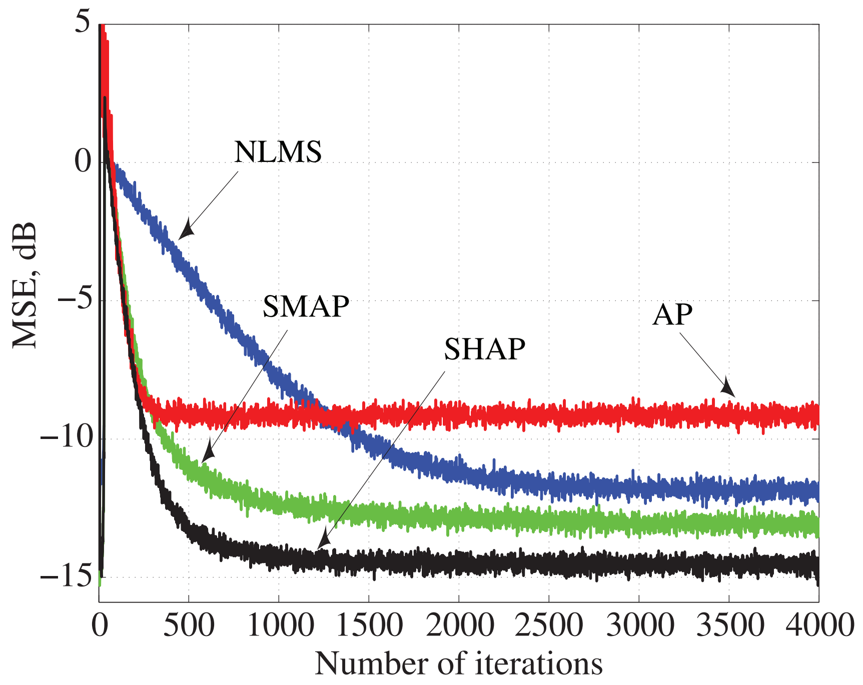

Figure 8.4: Learning curves with $L = 3$, $t = \sqrt{L\sigma_v^2}$.

Parameter λ in (8.17) was set to 0.99. The MSD curves obtained are illustrated in Fig. 8.6. As can be seen, the NPNLMS and SHNLMS algorithms offer similar but improved performance as compared to the other algorithms.

Next we carried out two experiments using the SHLMS algorithm with $M = 31$ and $f_c = 0.4$. In both experiments the measurement noise was a zero-mean white Gaussian noise signal with variance 10^{-3}. In the seventh experiment, the input signal was a zero-mean white Gaussian noise signal with variance 1. The MSD curves obtained by using the LMS and the SHLMS algorithms are illustrated in Fig. 8.7. As can be seen, the proposed SHLMS algorithm yields a reduced misalignment for the same convergence speed and a fast convergence for similar misalignment. In the eighth experiment, we used $M = 16$, $f_c = 0.4$, and the same input signal except that it was correlated by using an IIR filter with a single pole at 0.9. The MSD curves obtained by using the LMS, SHLMS, and the variable step size LMS (VLMS) algorithm reported in [6] are illustrated in Fig. 8.8. The parameters for the VLMS algorithm were set to $\mu_{min} = 0.00001$, $\mu_{max} = 0.001$, $\alpha = 0.999$, $\beta = 0.992$, $\gamma = 0.001$, and $q/E[\|\boldsymbol{x}_k\|^2]$ for the SHLMS algorithm was set to $1/300$. As can be seen, the proposed SHLMS algorithm converges faster than the other algorithms and at the same time it yields

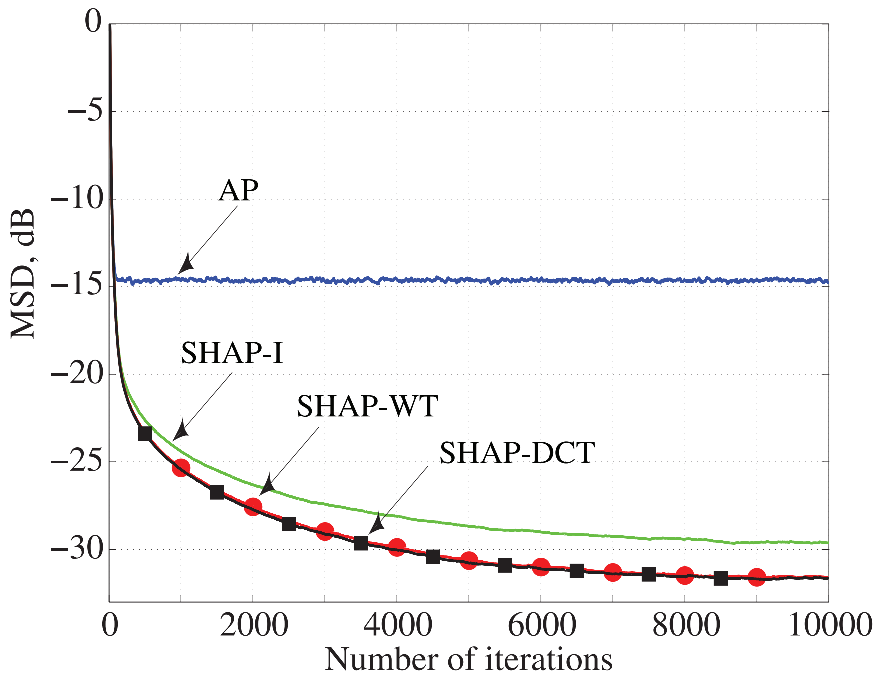

Figure 8.5: Learning curves with $L = 8$, $t = \sqrt{L\sigma_v^2}$.

a reduced steady-state misalignment.

8.5.2 Acoustic Echo-Cancelation Application

In this section, an acoustic-echo cancelation application is considered. The length of the acoustic channel was set to 1024. The loudspeaker was set to introduce a gain of 10^3 to the speech signal and the speech signal was contaminated with a zero-mean white Gaussian noise signal with variance 10^{-2}. The variance of the measurement noise added to the desired signal was set to 10^{-2}. The MSD curves obtained by using the NLMS, AP, VSSAP, and SHAP algorithms are illustrated in Fig. 8.9. As can be seen, the SHAP algorithm yields a reduced steady-state misalignment for the same convergence speed as compared to the VSSAP algorithm, and a faster convergence for similar steady-state misalignment as compared to the AP algorithm.

8.5.3 Identification of an IIR Filter

In this section, results obtained using a first-order IIR filter as the unknown system are presented. Four experiments were carried out with the pole of the IIR filter set

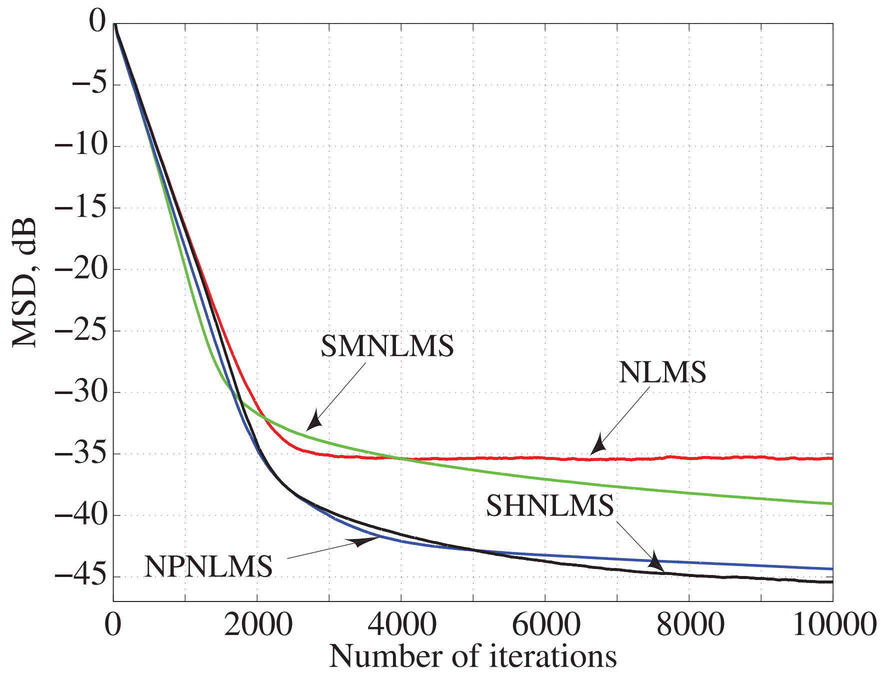

Figure 8.6: Learning curves with $L = 1$, $t = \sqrt{L\sigma_v^2}$.

to $\rho = 0.5,\ 0.7,\ 0.85$, and 0.9, and the length of the FIR filter set to M=27, 37, 111, and 63, respectively. The learning curves obtained are illustrated in Figs. 8.10–8.13, respectively. As can be seen, the SHAP algorithm outperforms the other algorithms for the same convergence speed but the SMAP algorithm is sensitive to the fact that an IIR filter cannot be modeled exactly by an FIR filter. This is because the error bound $\gamma = \sqrt{5}\sigma_v$ in the SMAP algorithm performs well only when exact modeling of the unknown system in terms of an FIR filter is possible.

8.6 Conclusions

A family of shrinkage adaptation algorithms, namely, the shrinkage AP, shrinkage NLMS, and shrinkage LMS algorithms were proposed based on the framework of iterative shrinkage/threshold method. In view of their advantages, the proposed shrinkage algorithms can provide improved solutions in many recent adaptive-filtering applications. The proposed algorithms were applied to system-identification and echo-cancelation applications. The simulation results obtained show that the SHAP algorithm performs much better than the conventional AP, SMAP, and VSSAP algo-

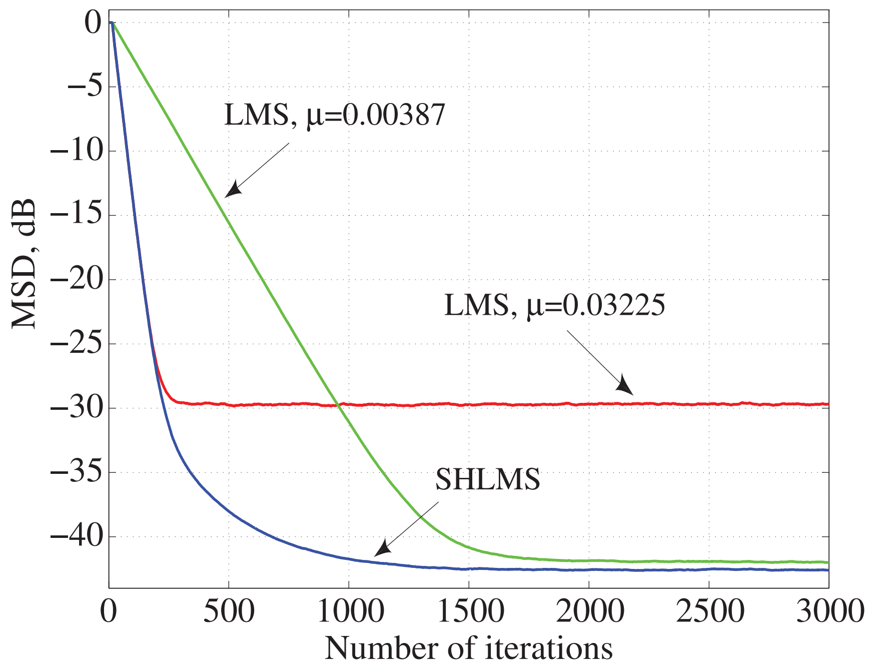

Figure 8.7: Learning curves with $\lambda = 0.95$, $q = 1$, and $t = \sqrt{2\sigma_v^2}$.

rithms in terms of steady-state misalignment and convergence speed. The SHNLMS algorithm offers a reduced steady-state misalignment as compared to the NLMS and SMNLMS algorithms. The SHLMS algorithm, on the other hand, offers faster convergence and a reduced steady-state misalignment as compared to the LMS and the VLMS algorithms.

The next chapter summarizes the conclusions of this dissertation and outlines some future research directions in the area of adaptive filtering.

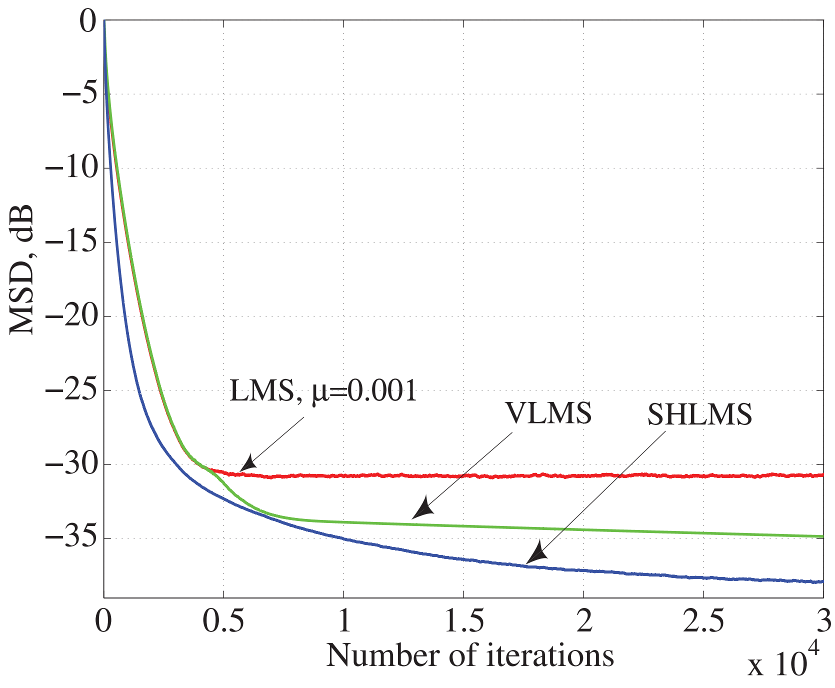

Figure 8.8: Learning curves with $\lambda = 0.999$ and $t = \sqrt{2\sigma_v^2}$.

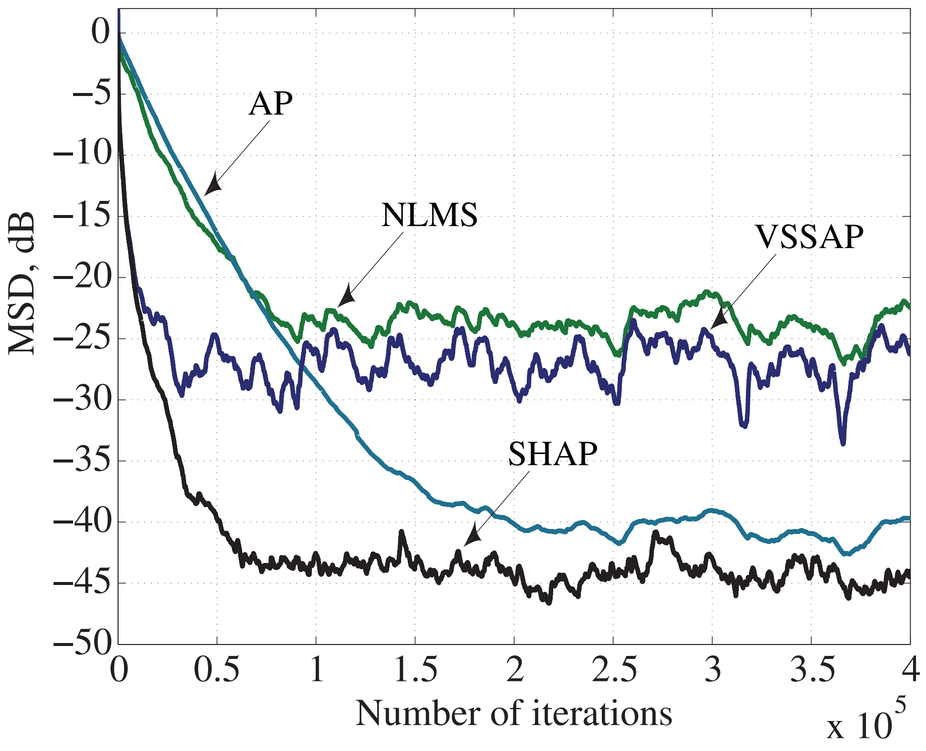

Figure 8.9: Learning curves with $L = 10$, $t = \sqrt{L\sigma_v^2}$.

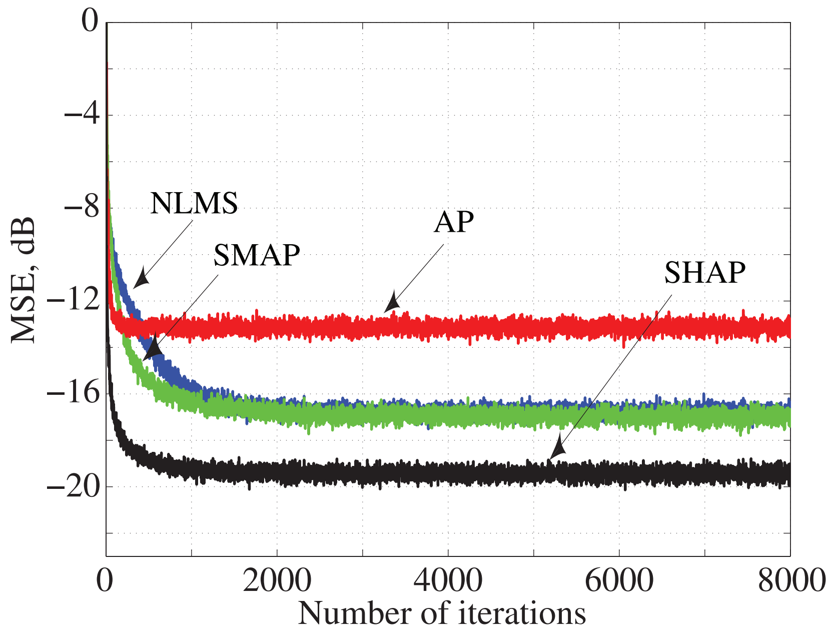

Figure 8.10: Learning curves with $L = 2$, $t = \sqrt{L\sigma_v^2}$.

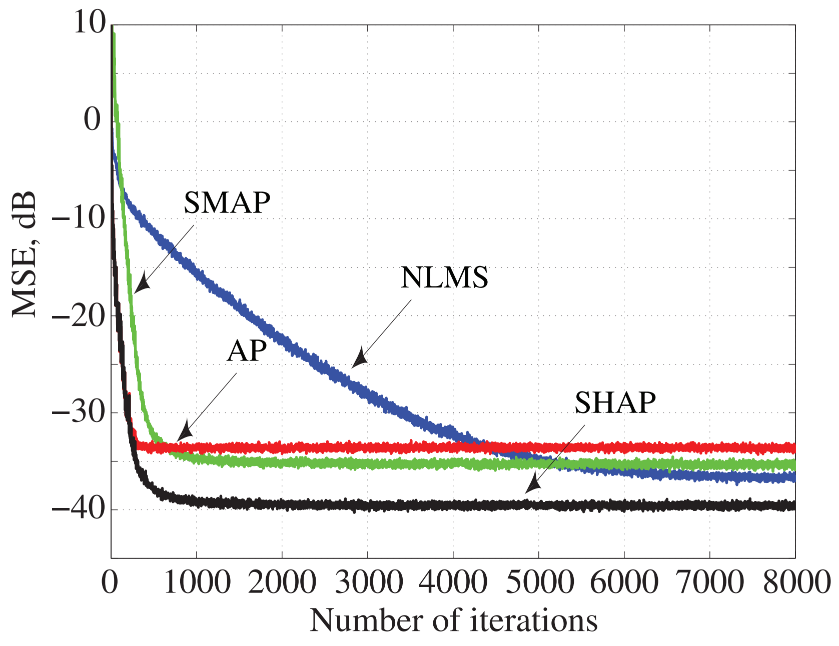

Figure 8.11: Learning curves with $L = 4$, $t = \sqrt{L\sigma_v^2}$.

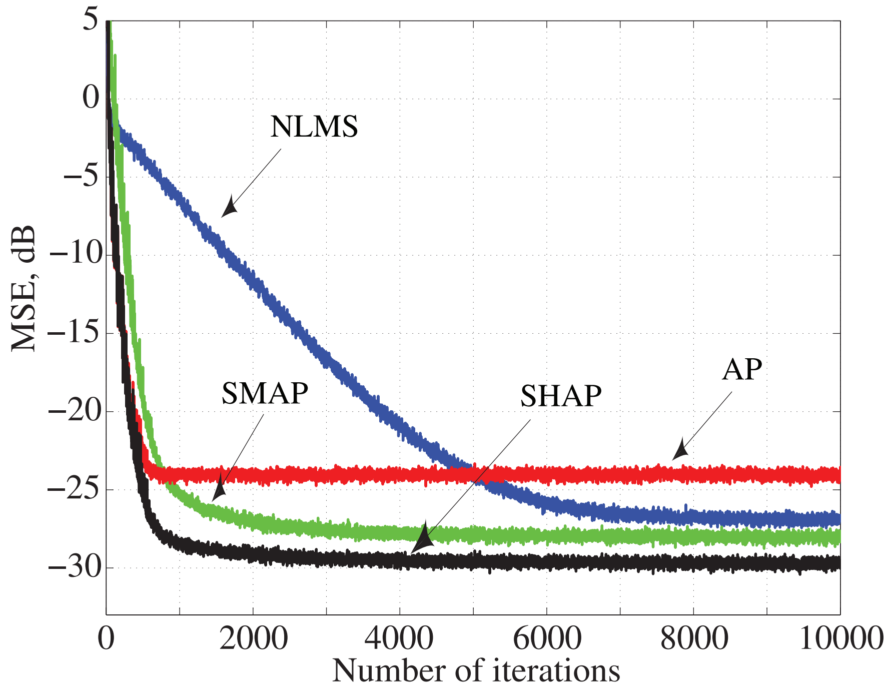

Figure 8.12: Learning curves with $L = 6$, $t = \sqrt{L\sigma_v^2}$.

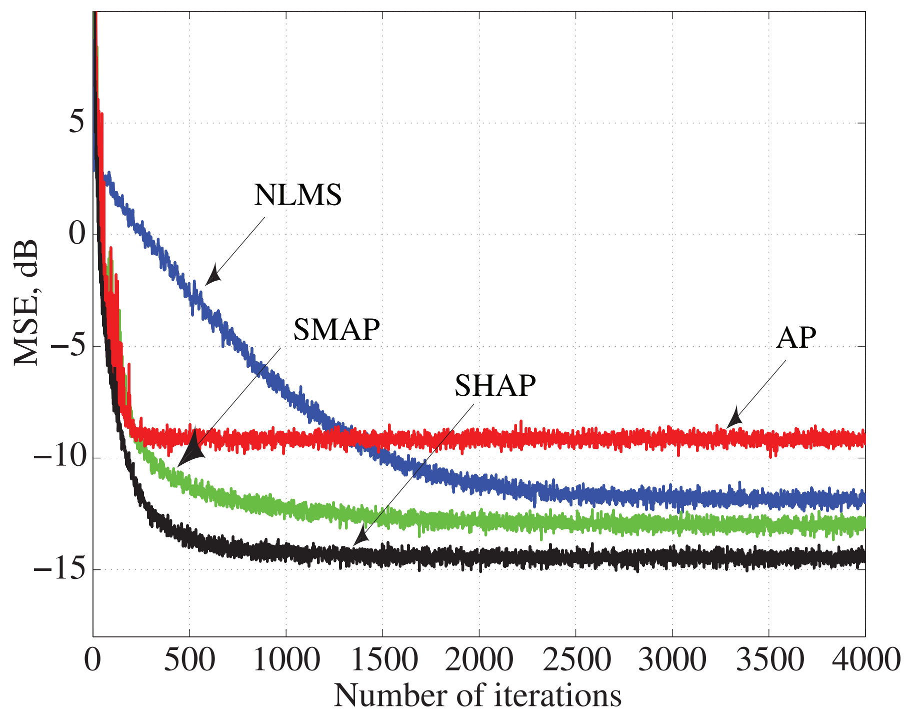

Figure 8.13: Learning curves with $L = 3$, $t = \sqrt{L\sigma_v^2}$.

Chapter 9

Conclusions and Recommendations for Future Research

9.1 Introduction

Several adaptation algorithms that offer improved performance in terms of robustness with respect to impulsive noise, tracking capability, convergence speed, steady-state misalignment, and computational load were proposed. The new algorithms are based on the set-membership affine projection (SMAP), constrained set-membership affine projection (CSMAP), recursive least-squares (RLS), quasi-Newton (QN), and minimum-error-entropy (MEE) adaptation algorithms. Stability and steady-state mean square-error (MSE) analyses were carried out. Extensive MATALB simulation results for a variety of applications were used to demonstrate the improved performance of the proposed algorithms and to verify the accuracy of the analytical results presented.

9.2 Conclusions

In chapter 2, a robust SMAP (RSMAP) algorithm was described that yields a significantly reduced steady-state misalignment, robust performance with respect to impulsive noise, and improved convergence speed as well as re-adaptation capability compared to those of the affine projection (AP) and SMAP algorithms. Two versions of the RSMAP algorithm were described. A steady-state MSE analysis of the proposed RSMAP algorithms was also carried out. Simulation results in acoustic-echo and network-echo cancelation applications showed that the RSMAP algorithms offer robust performance with respect to impulsive noise while retaining good tracking as compared to the AP, SMAP, and SMPAP algorithms. The steady-state MSE values obtained in a system-identification application matched quite well the values obtained using the derived expression for the steady-state MSE.

In chapter 3, a constrained version of the RSMAP algorithm was described that also yields a significantly reduced steady-state misalignment, robust performance with respect to impulsive noise, good convergence speed and re-adaptation capability compared to the CAP and CSMAP algorithms. Simulation results in linear-phase system-identification, time-series-filtering, and interference suppression applications showed that the proposed constrained SMAP (PCSMAP) algorithm offers a significantly reduced steady-state misalignment as compared to the conventional constrained AP and constrained SMAP algorithms.

In chapter 4, a new robust recursive least-square (RRLS) adaptation algorithm was proposed. The proposed RRLS (PRRLS) algorithm has two variants: one with a fixed forgetting factor and the other with a time-varying forgetting factor. The RRLS adaptation algorithm with a fixed forgetting factor was developed for applications in stationary environments. On the other hand, the PRRLS algorithm with a time-varying forgetting factor was developed for applications in nonstationary environments. Simulation results in a system-identification application in stationary environments showed that the PRRLS algorithm offers robust performance with respect to impulsive noise as compared to the conventional RLS and recursive least-mean (RLM) adaptation algorithms. For nonstationary environments, on the other hand, the proposed robust RLS algorithm offers robust performance with respect to impulsive noise as well as good tracking.

In chapter 5, an improved QN adaptation algorithm based on the classical QN optimization algorithm was introduced. The proposed QN (PQN) algorithm offers improved performance relative to that achieved with the known quasi-Newton (KQN) adaptation algorithm. Analyses have shown that the PQN algorithm is asymptotically stable and, furthermore, it yields a reduced steady-state misalignment relative to the known QN adaptation algorithm. Simulation results in a system-identification application showed that the PQN algorithm offers superior convergence speed as compared to the KQN algorithm for medium and high SNRs in a system-identification application. In addition, the PQN algorithm yields a reduced steady-state misalignment for low, medium, and high SNRs, which is as expected from the steady-state

MSE analysis. The steady-state MSEs obtained from the simulation results match quite well those obtained from the analytical expression of the steady-state MSE in both stationary and nonstationary environments.

In chapter 6, a robust version of the PQN (PRQN) algorithm for applications in impulsive noise environments was developed. Robustness with respect to impulsive noise is achieved by using two error bounds as was done in chapter 2. Switching between the two error bounds is done by using a variance estimator. For applications where the error bound cannot be easily estimated, a modified variance estimator can be used. Analysis shows that the PRQN is asymptotically stable. A steady-state MSE analysis of the PRQN algorithm was also carried out by using the energy-conservation principle described in [87]. Simulation results in a system-identification application showed that the PRQN algorithm yields robust performance with respect to impulsive noise and better tracking compared to the KQN and KRQN algorithms both in stationary and nonstationary environments. The steady-state MSEs obtained from the simulations match reasonably well the steady-state MSEs obtained using the analytical expression of the steady-state MSE of the PRQN algorithms.

In chapter 7, we developed a new normalized minimum-error entropy (PNMEE) algorithm. Like the stochastic minimum-error entropy (MEE), variable step size MEE (VMEE), and normalized MEE (NMEE) adaptation algorithms, the PNMEE algorithm is based on the minimum-error entropy criterion. The PNMEE algorithm has a reduced computational complexity and offers better performance than the MEE, VMEE, and NMEE algorithms. Simulation results in system-identification and signal-prediction applications showed that the PNMEE algorithm converges much faster than the MEE, NLMS, VMEE, NMEE algorithms and at the same time it yields a reduced steady-state misalignment. In addition, the convergence speed of the PNMEE algorithm does not depend on the input-signal power as is the case in the MEE and VMEE algorithms.

In chapter 8, a family of adaptation algorithms based on the iterative/shrinkage method [80, 81] has been proposed. The algorithms developed were the shrinkage LMS (SHLMS), shrinkage NLMS (SHNLMS), shrinkage AP (SHAP) adaptation algorithms and they were found to offer better performance compared to other competing algorithms. Simulation results in system-identification and acoustic-echo-cancelation applications showed that the proposed SHLMS, SHNLMS, and SHAP algorithms yield significantly reduced steady-state misalignments compared to the {LMS, VSSLMS}, {NLMS, SMNLMS, NPLMS}, and {AP, SMAP, VSSAP} algorithms, respectively, for the same convergence speed. Simulation results also showed that the steady-state MSEs of the SHAP algorithm do not depend on value of the projection order L.

For applications that entail impulsive noise, the PRQN algorithm in chapter 7 offers better performance compared to the RSMAP, PRRLS, and PQN, algorithms developed in chapters 2, 4, and 5, respectively. However, the computational complexity of the PRQN algorithm is larger than that of the RSMAP algorithm. On the other hand, the PRRLS algorithm in chapter 4 is easier to implement compared to the PRQN algorithm. However, the computation load of the PQN and PRQN algorithms is less than that of the PRRLS algorithm. For applications that do not entail impulsive noise, the SHAP algorithm of chapter 8 would outperform the RSMAP algorithm of chapter 2. In addition, the implementation of the SHAP algorithm is easier compared to that of the RSMAP algorithm. However, the computational load of the RSMAP algorithm is less than that of the SHAP algorithm. Finally, the SHLMS algorithm has computational complexity of the order of $\mathcal{O}(M)$ which is lower than those of all the other algorithms proposed.

9.3 Recommendations for Future Research

Further research on adaptation algorithms would be worthwhile on several fronts as detailed below.

The steady-state MSE analyses of the RSMAP, PQN, and PRQN algorithms in chapters 2, 5, and 6 involve certain simplifying assumptions and hence are approximate. A possible future effort would be to derive more exact formulas for the steady-state MSE without using the simplifying assumptions. Also it would be worthwhile to carry out a transient analysis for the RSMAP, PQN, and PRQN algorithms. Analyses pertaining to the steady-state MSE and transient behavior of the PRRLS algorithm of chapter 4 and of the SHLMS, SHNLMS, and SHAP algorithms of chapter 8 would be worthwhile to pursue.

The PRRLS adaptation algorithm in chapter 4 bounds the L_1 norm of the cross-correlation vector. It would also be interesting to bound other norms in order to obtain a robust RLS algorithm with respect to impulsive noise.

It should also be possible to apply the error-bound estimation technique of chapter 2 to the algorithms of chapter 8. If this could be done, then the algorithms of chapter 8 could be used in applications where information about the noise variance is not easy to obtain.

A constrained version of the SHLMS, SHNLMS, and SHAP algorithms could be developed following the approach used to develop a constrained version of the RSMAP algorithm of chapter 2, as was done in chapter 3.

The SHLMS, SHNLMS, and SHAP algorithms in chapter 8 are not robust with respect to impulsive noise. Therefore, there is considerable scope in developing robust versions of these algorithms for applications that entail impulsive noise.

The iterative/shrinkage method used in chapter 8 can also be used to develop shrinkage RLS and QN algorithms. This method is based on an L_1–L_2 minimization problem and, therefore, the solution obtained is very sparse. The iterative/shrinkage method was used to obtain a noise-free error signal in chapter 8. However, it can also be used to obtain a noise-free gain vector which would also be very sparse. By using a sparse gain vector, partial-update SHLMS, SHNLMS, and SHAP algorithms could be obtained. Algorithms of this type are useful in acoustic-echo-cancelation applications. The iterative/shrinkage method can also be used to obtain block LMS, NLMS, and AP algorithms [2] using a similar approach to that used to obtain the algorithms of chapter 8.

Bibliography

[1] S. Haykin, *Adaptive Filter Theory.* NewJersy: Prentice-Hall, 1996.

[2] A. H. Sayed, *Fundamentals of Adaptive Filtering.* NewJersy: Wiley, 2003.

[3] P. S. R. Diniz, *Adaptive Filtering: Algorithms and Practical Implementation.* NewYork: Springer, 3rd ed., 2008.

[4] B. Widrow and M. E. Hoff, "Adaptive switching circuits," *IRE Western Electric Show and Convention Record*, pp. 96–104, Aug. 1960.

[5] R. H. Kwong and E. W. Johnston, "A variable step size LMS algorithm," *IEEE Trans. Signal Process.*, vol. 40, pp. 1633–1642, July 1992.

[6] T. Aboulnasr and K. Mayyas, "A robust variable step size LMS-type algorithm: analysis and simulations," *IEEE Trans. Signal Process.*, vol. 45, no. 3, pp. 631–639, Mar. 1997.

[7] D. I. Pazaitis and A. G. Constantinides, "A novel kurtosis driven variable step-size adaptive algorithm," *IEEE Trans. Signal Process.*, vol. 47, pp. 864–872, Mar. 1999.

[8] J. I. Nagumo and A. Noda, "A learning method for system identification," *IEEE Trans. Automatic Control*, vol. AC-12, pp. 282–287, Jun. 1967.

[9] D. T. M. Slock, "On the convergence behavior of the LMS and the normalized LMS algorithms," *IEEE Trans. Signal Procsss.*, vol. 41, no. 9, pp. 2811–2825, Sep. 1993.

[10] J. Benesty, H. Rey, L. R. Vega, and S. Tressens, "A nonparametric VSS NLMS algorithm," *IEEE Signal Process. Letters*, vol. 13, no. 10, pp. 581–584, Oct. 2006.

[11] K. Ozeki and T. Umeda, “An adaptive filtering algorithm using an orthogonal projection to an affine subspace and its properties,” *Electron. Commun. Japan*, vol. 67-A, pp. 19–27, 1984.

[12] S. G. Kraizer and D. R. Morgan, “The partial-rank algorithm for adaptive beamforming,” in *Proc. SPIE Int. Soc. Opt. Eng.*, vol. 564, pp. 9–14, 1985.

[13] D. T. M. Slock, “The block underdetermined covariance BUC fast transversal filer FTF algorithm for adaptive filtering,” in *Proc. Asilomar Conf. Signals, Systems and Computers*, pp. 550–554, Oct. 1992.

[14] S. L. Gay and S. Tavathia, “The fast affine projection algorithm,” in *Proc. Intern. Conf. Acoustic, Speech and Signal Process.*, pp. 3023–3026, May 1995.

[15] M. Montazeri and P. Duhamel, “A set of algorithm linking NLMS and block RLS algorithms,” *IEEE Trans. Signal Process.*, vol. 46, no. 3, pp. 771–775, Mar. 1998.

[16] M. Rupp, “A family of adaptive filtering algorithms with decorrelating properties,” *IEEE Trans. Signal Process.*, vol. 46, no. 3, pp. 771–775, Mar. 1998.

[17] S. L. Gay and J. Benesty, *Acoustic Signal Processing for Telecommunication.* Boston: Springer, 2000.

[18] S. G. Sankaran and A. A. Beex, “Convergence behavior of affine projection algorithms,” *IEEE Trans. Signal Process.*, vol. 48, no. 4, pp. 1086–1096, Apr. 2000.

[19] S. Werner and P. S. R. Diniz, “Set-membership affine projection algorithm,” *IEEE Signal Process. Letters*, vol. 8, no. 8, pp. 231–235, Aug. 2001.

[20] H. C. Shin and A. H. Sayed, “Mean-square performance of a family of affine projection algorithms,” *IEEE Trans. on Signal Process.*, vol. 52, no. 1, pp. 90–102, Jan. 2004.

[21] H. C. Shin, A. H. Sayed, and W. J. Song, “Variable step-size NLMS and affine projection algorithms,” *IEEE Signal Process. Letters*, vol. 11, no. 2, pp. 132–135, Feb. 2004.

[22] H. Rey, L. R. Vega, S. Tressens, and J. Benesty, “Variable explicit regularization in affine projection algorithm: Robustness issues and optimal choice,” *IEEE Trans. Signal Process.*, vol. 55, no. 5, pp. 2096–2109, May 2007.

[23] C. Paleologu, B. J, and S. Ciochina, “A variable step-size affine projection algorithm designed for acoustic echo cancelation,” *IEEE Trans. Audio, Speech, and Language Process.*, vol. 16, no. 8, pp. 1466–1478, Nov. 2008.

[24] M. Chang, N. Kong, and P. Park, “An affine projection algorithm based on reuse time of input vectors,” *IEEE Signal Process. Letters*, vol. 7, no. 8, pp. 750–753, Aug. 2010.

[25] S. Werner, J. A. Apolinario Jr., P. S. R. Diniz, and T. I. Laakso, “Set-membership approach to normalized proportionate adaptation algorithms,” in *Proc. Europ. Signal Process. Conf.*, Sep. 2005.

[26] S. Werner, P. S. R. Diniz, and J. E. W. Moreira, “Set-membership affine projection algorithm with variable data-reuse factor,” in *Proc. IEEE Intern. Symp. Circuits and Syst.*, pp. 261–264, May 2006.

[27] S. Werner, J. A. Apolinario, M. L. R. De Campos, and P. S. R. Diniz, “Low-complexity constrained affine projection algorithms,” *IEEE Trans. Signal Process.*, vol. 53, no. 12, pp. 4545–4555, Dec. 2005.

[28] O. L. Frost III, “An algorithm for linearly constrained adaptive array processing,” *Proceedings of the IEEE*, vol. 60, pp. 926–935, Aug. 1972.

[29] L. J. Griffiths and C. W. Jim, “An alternative approach to linearly constrained adaptive beamforming,” *IEEE Trans. Antennas propagation*, vol. AP-30, pp. 27–34, Jan. 1982.

[30] M. Honig, U. Madhow, and S. Verdu, “Blind adaptive multiuser detection,” *IEEE Trans. Info. Theory*, vol. 41, pp. 944–960, Jul. 1985.

[31] S. C. Park and J. F. Doherty, “Generalized projection algorithm for blind interference suppression in DS/CDMA communications,” *IEEE Trans. Circuit Syst.- Part II*, vol. 44, pp. 453–460, Jun. 1997.

[32] J. A. Apolinario, S. Werner, and P. S. R. Diniz, “Constrained normalized adaptive filters for CDMA mobile communications,” in *Proc. Euro. Signal Process. Conf.*, vol. 4, pp. 2053–2056, 1998.

[33] S. Nagraj, S. Gollamudi, S. Kapoor, Y. F. Huang, and J. R. Deller, "Adaptive interference suppression for CDMA systems with a worst-case error criterion," *IEEE Trans. Signal Process.*, vol. 48, no. 1, pp. 284–289, Jan. 2000.

[34] S. Gullamudi, S. Nagaraj, S. Kapoor, and Y. -F. Huang, "Set-membership filtering and a set membership normalized LMS algorithm with an adaptive step size," *IEEE Signal Proces. Letters*, vol. 5, pp. 111–114, Apr. 1998.

[35] P. S. R. Diniz and S. Werner, "Set-membership binormalized data-reusing LMS algorithms," *IEEE Trans. Signal Process.*, vol. 51, no. 1, pp. 124–134, Jan. 2003.

[36] K. Tsakalis, M. Deisher, and A. Spanias, "System identification based on bounded error constraints," *IEEE Trans. Signal Process.*, vol. 43, no. 12, pp. 3071–3075, Dec. 1995.

[37] E. Fogel and Y. Huang, "On the value of information in system identification–Bounded noise case," *Automatica*, vol. 18, p. 229, 1982.

[38] V. Atti, A. Spanias, K. Tsakalis, C. Panayiotou, L. Iasemidis, and V. Berisha, "Gradient projection-based channel equalization under sustained fading," *Signal Processing*, vol. 88, pp. 236–246, 2008.

[39] P. S. R. Diniz, "Convergence performance of the simplified set-membership affine projection algorithm," *Circuits, Systems, and Signal Process.*, vol. 30, no. 2, pp. 439–462, Nov. 2010.

[40] M. Z. A. Bhotto and A. Antoniou, "A set-membership NLMS algorithm with adaptive error bound," in *Proc. IEEE Can. Conf. on Elect. and Comp. Eng.*, pp. 2007–2010, Jun. 2008.

[41] M. Z. A. Bhotto and A. Antoniou, "A set-membership affine projection algorithm with adaptive error bound," in *Proc. IEEE Can. Conf. on Elect. and Comp. Eng.*, pp. 894–897, May 2009.

[42] M. Z. A. Bhotto and A. Antoniou, "A robust set-membership normalized least mean-square adaptive filter," in *Proc. IEEE Can. Conf. on Elect. and Comp. Eng.*, May 2010.

[43] L. Jung, M. Morf, and D. Flaconer, “Fast calculations of gain matrices for recursive estimation schemes,” *Intern. Journal Contr.*, vol. 27, pp. 1–19, Jan. 1978.

[44] G. Carayannis, D. Manolakis, and N. Kalouptsidis, “A fast sequential algorithm for least-squares filtering and prediction,” *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 31, no. 6, pp. 1394–1402, Dec. 1983.

[45] J. Coiffi and T. Kailath, “Fast recursive LS transversal filters for adaptive processing,” *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 32, no. 4, pp. 304–337, Apr. 1984.

[46] G. Carayannis, D. Manolakis, and N. Kalouptsidis, “A unified view of parametric processing algorithms for prewindowed signals,” *Signal Process.*, vol. 10, pp. 335–368, 1986.

[47] N. Kalouptsidis and S. Theodoridis, *Efficient System Identification and Signal Processing Algorithms.* Englewood cliffs, NJ: Prentice-Hall, 1993.

[48] J. M. Cioffi, “The fast adaptive rotor’s RLS algorithm,” *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 38, no. 4, pp. 631–651, Apr. 1990.

[49] P. A. Regalia and M. G. Bellanger, “On the duality between fast QR methods and lattice methods in least squares adaptive filtering,” *IEEE Trans. Signal Process.*, vol. 39, no. 4, pp. 879–892, Apr. 1991.

[50] B. Yang and J. F. Bohme, “Rotation-based RLS algorithms: Unified derivations, numerical properties and parallel implementation,” *IEEE Trans. Signal Process.*, vol. 40, no. 5, pp. 1151–1167, May. 1992.

[51] S. T. Alexander and A. Ghirnikar, “A method for recursive least squares filtering based upon an inverse qr decomposition,” *IEEE Trans. Signal Process.*, vol. 41, no. 1, pp. 20–30, Jan. 1993.

[52] D. Slock, “Reconciling fast RLS lattice and QR algorithms,” in *Proc. IEEE Intern. Conf. on Acoust., Speech, and Signal Process.*, pp. 1591–1594, Apr. 1990.

[53] A. A. Rontogiannis and S. Theodoridis, "New fast QR least squares adaptive algorithms," in *Proc. IEEE Intern. Conf. on Acoust., Speech, and Signal Process.*, pp. 1412–1415, May 1995.

[54] I. K. Proudler, J. G. McWhirter, and T. J. Shepherd, "Fast QRD-based algorithms for least squares linear prediction," in *Proc. IMA Conf. Math. Signal Process.*, (Warwick, England).

[55] G. E. Bottomley and S. T. Alexander, "A novel approach for stabilizing recursive least squares filters," *IEEE Trans. Signal Process.*, vol. 39, no. 8, pp. 1770–1779, Aug. 1991.

[56] A. P. Liavas and P. A. Regalia, "On the numerical stability and accuracy of the conventional recursive least squares algorithm," *IEEE Trans. Signal Process.*, vol. 47, no. 1, pp. 88–96, Jan. 1999.

[57] M. L. R. De Campos and A. Antoniou, "A robust quasi-Newton adaptive filtering algorithm," in *Proc. IEEE Intern. Symp. Circuits Syst.*, pp. 229–232, Nov. 1994.

[58] M. L. R. De Campos and A. Antoniou, "A new quasi-Newton adaptive filtering algorithm," *IEEE Trans. Circuits and Syst.*, vol. 44, no. 11, pp. 924–934, Nov. 1997.

[59] P. S. R. Diniz, M. L. R. De Campos, and A. Antoniou, "Analysis of LMS-Newton adaptive filtering algorithms with variable convergence factor," *IEEE Trans. on Signal Process.*, vol. 43, pp. 617–627, Mar. 1995.

[60] M. Z. A. Bhotto and A. Antoniou, "Improved data-selective LMS-Newton adaptation algorithms," in *Proc. Intern. Conf. on Digital Signal Process.*, Jul. 2009.

[61] S. Koike, "Adaptive threshold nonlinear algorithm for adaptive filters with robustness against impulsive noise," *IEEE Trans. Signal Process.*, vol. 45, no. 9, pp. 2391–2395, Sep. 1997.

[62] J. F. Weng and S. H. Leung, "Adaptive nonlinear RLS algorithm for robust filtering in impulse noise," in *Proc. IEEE Inter. Symp. Circuits Syst.*, pp. 2337–2340, Jun. 1997.

[63] Y. Zou, S. C. Chan, and T. S. Ng, “A recursive least M-estimate (RLM) adaptive filter for robust filtering in impulsive noise,” *IEEE Signal Process. Letters*, vol. 7, no. 11, pp. 324–326, Nov. 2000.

[64] Y. Zou and S. C. Chan, “A robust quasi-Newton adaptive filtering algorithm for impulse noise suppression,” in *Porc. IEEE Intern. Symp. Circuits Syst.*, pp. 677–680, May 2001.

[65] P. Petrus, “Robust huber adaptive filter,” *IEEE Trans. Signal Process.*, vol. 47, no. 4, pp. 1129–1133, Apr. 1999.

[66] P. J. Rousseeuw and A. M. Leroy, *Robust regression and outlier detection.* New York: Wiley, 1987.

[67] S. C. Chan and Y. Zou, “A recursive least M-estimate algorithm for robust adaptive filtering in impulsive noise: Fast algorithm and convergence analysis,” *IEEE Trans. Signal Process.*, vol. 52, no. 4, pp. 975–991, Apr. 2004.

[68] L. R. Vega, H. Rey, J. Benesty, and S. Tressens, “A fast robust recursive least-squares algorithm,” *IEEE Trans. Signal Process.*, vol. 57, no. 3, pp. 1209–1215, Mar. 2009.

[69] M. L. R. De Campos, *Development and analysis of fast and robust Newton-type adaptation algorithms.* Victoria, Canada: Ph.D. Dissertation, University of Victoria, 1995.

[70] A. Antoniou and W. S. Lu, *Practical Optimization.* USA: Springer, 2007.

[71] R. Fletcher, *Practical Methods of Optimization.* New Jersey: Wiley, 1980.

[72] M. L. R. de Campos, S. Werner, J. A. Apolinario Jr., and P. S. R. Diniz, “Constrained quasi-Newton algorithm for CDMA mobile communications,” in *Proc. Inter. Telecomm. Symp.*, pp. 371–376, Aug. 1998.

[73] D. F. Marshall and W. K. Jenkins, “A fast quasi-Newton adaptive filtering algorithm,” *IEEE Trans. Signal Process.*, vol. 7, no. 40, pp. 1652–1662, Jul. 1992.

[74] D. Erdogmus and J. C. Pŕincipe, “An entropy minimization algorithm for supervised training of nonlinear systems,” *IEEE Trans. Signal Process.*, vol. 50, no. 7, pp. 1780–1786, Jul. 2002.

[75] D. Erdogmus and J. C. Píncipe, "Generalized information potential criterion for adaptive system training," *IEEE Trans. Neural Networks*, vol. 13, no. 5, pp. 1035–1044, Sep. 2002.

[76] D. Erdogmus, K. E. Hild, and J. C. Píncipe, "Online entropy manipulation: stochastic information gradient," *IEEE Signal Process. Letters*, vol. 10, no. 8, Aug. 2003.

[77] S. Han, S. Rao, D. Erdogmus, and J. C. Píncipe, "An improved minimum error entropy criterion with self adjusting step-size," in *Proc. IEEE Workshop on Machine Learning for Signal Process.*, pp. 317–322, Sep. 2005.

[78] S. Han, S. Rao, K. Jeong, and J. C. Píncipe, "A normalized minimum error entropy stochastic algorithm," in *Proc. IEEE Intern. Conf. on Acoust., Speech, and Signal Process.*, pp. 609–612, Sep. 2006.

[79] A. Renyi, *Probability Theory.* New York: Elsevier., 1970.

[80] M. Zibulevsky and M. Elad, "l_1-l_2 optimization in signal and image processing," *IEEE Signal Process. Magazine*, pp. 76–88, May 2010.

[81] I. Daubechies, M. Defrise, and C. De-Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Comm. Pure Appl. Math.*, vol. 57, pp. 1413–1457, 2004.

[82] M. Z. A. Bhotto and A. Antoniou, "Robust set-membership affine-projection adaptive-filtering algorithm," *IEEE Trans. Signal Process.*, vol. 60, no. 1, pp. 73–81, Jan. 2012.

[83] M. Z. A. Bhotto and A. Antoniou, "Constrained robust set-membership affine-projection adaptive-filtering algorithm," in *Proc. IEEE Inter. Symp. Comm. Cont. and Signal Process.*, pp. x–x, May 2012. submitted.

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

[85] M. Z. A. Bhotto and A. Antoniou, "Improved quasi-Newton adaptive filtering algorithm," *IEEE Trans. Circuits Syst.-I*, vol. 57, no. 8, pp. 2109–2119, Aug. 2010.

[86] M. Z. A. Bhotto and A. Antoniou, "Robust quasi-Newton adaptive filtering algorithms," *IEEE Trans. Circuits and Syst.-II*, vol. 58, no. 8, pp. 537–541, Aug. 2011.

[87] A. H. Sayed and M. Rupp, "Error-energy bounds for adaptive gradient algorithms," *IEEE Trans. on Signal Process.*, vol. 44, no. 8, pp. 1982–1989, Aug. 1996.

[88] M. Z. A. Bhotto and A. Antoniou, "A new normalized minimum-error entropy algorithm with reduced computational complexity," in *Proc. IEEE Intern. Symp. Circuits Syst.*, pp. 2561–2564, May 2009.

[89] J. W. Xu, D. Erdogmus, M. C. Ozturk, and J. C. Prı́ncipe, "Recursive renyi's entropy estimator for adaptive filtering," in *Proc. IEEE Intern. Symp. on Signal Process. and Information Tech.*, pp. 134–137, Dec. 2003.

[90] M. Z. A. Bhotto and A. Antoniou, "A family of shrinkage adaptive filtering algorithms," *IEEE. Trans. Signal Process.*, 2011. submitted.

[91] L. Guo and Y. F. Huang, "Set membership adaptive filtering with parameter-dependent error bound tuning," in *Proc. Intern. Conf. Acoust. Speech, Signal Process.*, vol. 17, pp. 369–372, 2005.

[92] J. F. Galdino, J. A. Apolinario Jr., and M. L. R. de Campos, "A set-membership NLMS algorithm with time varying error bound," in *in Proc. IEEE Intern. Symp. Circuits Syst.*, pp. 277–280, 2006.

[93] Y. Zou, S. C. Chan, and T. S. Ng, "Least mean M-estimate algorithms for robust adaptive filtering in impulse noise," in *Proc. Europ. Signal Process. Conf.*, Finland 2000.

[94] Y. Zou, S. C. Chan, and T. S. Ng, "A robust statistics based adaptive lattice-ladder filter in impulsive noise," in *Proc. IEEE Intern. Symp. Circuits Syst.*, pp. 539–542, May 2000.

[95] N. R. Yousef and A. H. Sayed, "A unified approach to the steady-state and tracking analyses of adaptive filters," *IEEE Trans. on Signal Process.*, vol. 49, no. 2, pp. 314–324, Feb. 2001.

[96] T. Y. Al-Naffouri and A. H. Sayed, "Transient analysis of adaptive filters with error nonlinearities," *IEEE Trans. on Signal Process.*, vol. 51, no. 3, pp. 653–663, Mar. 2003.

[97] T. Y. Al-Naffouri and A. H. Sayed, "Transient analysis of data-normalized adaptive filters," *IEEE Trans. on Signal Process.*, vol. 51, no. 3, pp. 639–652, Mar. 2003.

[98] H. C. Shin, W. J. Song, and A. H. Sayed, "Mean-square performance of data-reusing adaptive algorithms," *IEEE Signal Process. Letters*, vol. 12, no. 12, pp. 851–854, Dec. 2005.

[99] L. S. Resende, J. M. T. Romano, and M. G. Bellanger, "A fast least-squares algorithm for linearly constrained adaptive filtering," *IEEE Trans. Signal Process.*, vol. 44, no. 5, pp. 1168–1174, May 1996.

[100] B. Peterson and K. Narendra, "Bounded error adaptive control," *IEEE Trans. Automatic Control*, vol. 27, no. 6, pp. 1161–1168, Dec. 1982.

[101] E. Eleftheriou and D. D. Falconer, "Tracking properties and steady-state performance of RLS adaptive filter algorithms," *IEEE Trans. Acoustic, Speech, and Signal Process.*, vol. ASSP-34, pp. 1097–1110, Oct. 1986.

[102] P. Z. Peebles, *Probability, Random Variables and Random Signal Principles.* New York: McGraw-Hill, 2000.

[103] D. Erdogmus, J. C. Príncipe, S.-P. Kim, and J. Sanchez, "A recursive renyi's entropy estimator," in *Proc. NNSP'02*, (Martigni, Switzerland), pp. 209–217, Jan. 2002.

[104] A. H. Sayed and C. G. Lopes, "Adaptive processing over distributed networks," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. 90, no. 8, pp. 1504–1510, Aug. 2007.

[105] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Trans. Signal Process.*, vol. 55, no. 8, pp. 4064–4077, Aug. 2007.

[106] C. G. Lopes and A. H. Sayed, “Diffusion least-mean squares over adaptive networks: Formulation and performance analysis,” *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3122–3136, Jul. 2008.

[107] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, “Diffusion recursive least-squares for distributed estimation over adaptive networks,” *IEEE Trans. Signal Process.*, vol. 56, no. 5, pp. 1865–1877, May 2008.

[108] F. S. Cattivelli and A. H. Sayed, “Diffusion LMS strategies for distributed estimation,” *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1035–1048, Mar. 2010.

VITA
Md. Zulfiquar Ali Bhotto

Institute attended	Rajshahi University of Engineering and Technology, Bangladesh
Degree awarded	BSc in Electrical and Electronic Engineering, 2002
Awards	Uvic Fellowship, The University of Victoria, 2007–2008

Journal papers

1 Md. Z. A. Bhotto and A. Antoniou, "Improved quasi-Newton adaptive filtering algorithm," *IEEE Trans. Circuits Syst.-I*, vol. 57, no. 8, pp. 2109–2118, Aug. 2010.

2 Md. Z. A. Bhotto and A. Antoniou, "Robust recursive least-squares adaptive-filtering algorithm for impulsive-noise environment," *IEEE Signal Process. Letters*, vol. 18, no. 3, pp. 185–188, Mar. 2011.

3 Md. Z. A. Bhotto and A. Antoniou, "Robust quasi-Newton adaptive filtering algorithms," *IEEE Trans. Circuits Syst.-II*, vol. 58, no. 8, pp. 537–541, Aug. 2011.

4 Md. Z. A. Bhotto and A. Antoniou, "Robust set-membership affine-projection adaptive-filtering algorithm," *IEEE Trans. Signal Process.*, vol. 60, no. 1, pp. 73–81, Jan. 2012.

5 Md. Z. A. Bhotto and A. Antoniou, "A family of shrinkage affine-projection adaptive-filtering algorithms," *IEEE Trans. Signal Process.* (submitted).

Conference papers

1 Md. Z. A. Bhotto and A. Antoniou, "A set-membership NLMS algorithm with adaptive error bound," *in Proc. IEEE Can. Conf. on Elect. and Comp. Eng.*, Niagara Falls, Canada, pp. 2007–2010, Jun. 2008.

2 Md. Z. A. Bhotto and A. Antoniou, “A set-membership affine projection algorithm with adaptive error bound,” *in Proc. IEEE Can. Conf. on Elect. and Comp. Eng.*, Newfoundland, Canada, pp. 894–897, May 2009.

3 Md. Z. A. Bhotto and A. Antoniou, “A new normalized minimum error entropy algorithm with reduced computational complexity,” *in Proc. IEEE Intern. Symp. Circuits and Syst.*, Taipe, Taiwan, pp. 2561–2564, May 2009.

4 Md. Z. A. Bhotto and A. Antoniou, “Improved data-selective LMS-Newton adaptation algorithms,” *in Proc. Intern. Conf. on Digital Signal Process.*, Santorini, Greece, Jul. 2009.

5 Md. Z. A. Bhotto and A. Antoniou, “A robust set-membership normalized least mean-square adaptive filter,” *in Proc. IEEE Can. Conf. on Elect. and Comp. Eng.* Calgary, Canada, May 2010.

6 Md. Z. A. Bhotto and A. Antoniou, “A robust constrained set-membership affine projection adaptive-filtering algorithm,” *IEEE Inter. Symp. Comm. Cont. and Signal Process.* (submitted).