FAST BLIND EQUALIZATION USING SUBGRADIENT-BASED ALGORITHMS

by

Can Kızılkale

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Electrical & Computer Engineering

Koç University

September, 2004

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Can Kızılkale

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

Assistant Prof. Alper T. Erdoğan

Prof. Murat Tekalp

Assistant Prof. Metin Türkay

Date: _____

To my parents

ABSTRACT

In this thesis, several novel blind equalization methods based on subgradient search over a convex cost surface are presented. These subgradient-based methods are alternatives to the existing iterative blind equalization approaches (such as the Constant Modulus Algorithm (CMA)) which mostly suffer from the convergence problems caused by their nonconvex cost functions. The proposed methods are variations of on an iterative algorithm (called SubGradient based Blind Algorithm (SGBA)) for both real and complex constellations. SGBA is based on the minimization of the l_∞ norm of the equalizer output under a linear constraint on the equalizer coefficients using subgradient iterations. The algorithm has a nice convergence behavior attributed to the convex l_∞ cost surface as well as the step size selection rules connected with the subgradient search. We study four different variations of the SGBA algorithm: Fixed Window SGBA, Moving Window SGBA, Weighted SGBA and the Fractionally-Spaced SGBA. The performances of these algorithms are illustrated using examples in both complex and real constellations, where it is shown that the convergence behaviors of the proposed algorithms are in general less sensitive to initial point selection and fast convergence speeds can be achieved with a wise selection of step sizes. Furthermore, the amount of data required for the training of these equalization algorithms and their complexities are significantly low.

ACKNOWLEDGMENTS

..... .

TABLE OF CONTENTS

List of Tables ix

List of Figures x

Nomenclature xi

Chapter 1: Introduction 1

1.1 Blind Equalization Setup 3

1.2 Problem Definition 5

1.3 Thesis Overview 6

Chapter 2: Mathematical Background 7

2.1 Convexity 7

2.1.1 Convex Set 7

2.1.2 Convex Function 8

2.2 Subgradient 9

2.3 The Subgradient Algorithm 10

2.3.1 Zero-Limit-Divergent-Sum (ZLDS) Step Size Rule 11

2.3.2 Relaxation Step Size Rule 12

2.3.3 Relaxation Step Size Rule With Variable Target Values 13

Chapter 3: Previous Work 16

3.1 Classification Of Blind Equalization Methods 16

3.1.1 Subspace Methods 16

3.1.2 Higher Order Statistics Methods 17

3.1.3 Maximum Likelihood Methods 17

3.2 Existing Popular Algorithms Using Non-Convex Cost Functions 18

3.2.1 CMA 18
3.2.2 Shalvi Weinstein 21
3.2.3 Super Exponential Method 21
3.3 Blind Equalization By Using Convex Cost Functions 23
3.3.1 Necessary Conditions For Convex Cost Functions 23
3.3.2 l_∞ Norm As A Convex Cost Function 24
3.3.3 l_p Norm As Convex Cost Function (Vembu Algorithm) 26
3.3.4 Linear Programming Based Methods 27

Chapter 4: Fast and Low Complexity Blind Equalization via Subgradient Projections 30

4.1 SGBA 30
4.1.1 Step Size Selection for SGBA 31
4.2 Fixed Window SGBA 33
4.3 Moving window SGBA 33
4.4 Weighted SGBA 35
4.5 Fractionally Spaced SGBA 37

Chapter 5: Complexity Analysis 41

5.1 Moving Window SGBA 42
5.2 Weighted SGBA 43
5.3 Fractionally Spaced SGBA 43

Chapter 6: Performance Analysis Of SGBA 44

6.1 Comparison Of Step Size Rules 44
6.2 Comparison of Blind Algorithms: Complex Channel 46
6.3 Comparison of Blind Algorithms : DSL Channel 48
6.4 Moving Window SGBA 50
6.5 Fractionally Spaced SGBA 52

Chapter 7: Conclusions 55

7.1 Future Work 56

Bibliography **57**

Vita **62**

LIST OF TABLES

3.1 Different Step Sizes For CMA 20

LIST OF FIGURES

1.1 Equalization Setup. 3
1.2 Fractionally Spaced Equalizer. 4

2.1 Convex and Non-convex Sets. 8
2.2 A convex function 9
2.3 The tangential plane formed by a subgradient at point x. 9

3.1 Convex and non-convex cost surfaces. 20

4.1 Cost function on an arbitrary line crossing from the origin. 38
4.2 In this case SGBA algorithm converges to the origin directly. 40

6.1 Distance to the optimal point for different step size selections. 44
6.2 Open eye measure for different step size selections.. 46
6.3 ISI as a function of iterations for the 4-tap complex channel. 47
6.4 Impulse Response and the Frequency Response for a Sample DSL Channel. . 49
6.5 ISI as a function of iterations for the sample DSL channel. 50
6.6 Fixed Window SGBA ISI as a function of iterations for the sample channel. . 51
6.7 Moving Window SGBA ISI as a function of iterations for the sample channel. 52
6.8 Fractionally spaced SGBA, delta function as the starting point. 53
6.9 Fractionally spaced SGBA, starting point chosen for converging to zero. . . . 54

NOMENCLATURE

x : Point x

$\mathbf{x}$: Vector x

$\mathbf{X}$: Matrix x

$\| \mathbf{x} \|_p$: The l_p norm of the vector $\mathbf{x}$.

$\| \mathbf{x} \|_{\prod}$: Weighted norm of vector x. $\prod$ is the weighting matrix.

Co : Convex Hull.

$\partial f(x)$: Subdifferential of function f at point x.

Chapter 1

INTRODUCTION

Communications has always been a major part of human life. It is one of the few areas that almost every new technology is used in. In the near past, there has been a drastic improvement in communications. Especially the introduction of wireless technology redefined the way of communications. Not only the speed but also the mobility for communications have increased.

Today from internet to cell phones we face with communications in every aspect of life. To service the increasing need for the communications we need to use the available bandwidth wisely and make the data rate as high as possible. Also as the need increases new problems arise. One of the most important problems is the effect of the medium that the transmitted signal is passing through. For example as the signal passes through air it is reflected through different objects and several delayed, phase shifted and scaled versions of the signal reach the receiver. To use the information in these signals first we have to get rid of the effects caused by the media. This process is called equalization. Today most of the communication types requires this process. The equalization process must be fast and reliable since we are using high data rate. Standard equalization processes makes use of the training data which consumes the precious bandwidth. To get rid of the problems caused by training data blind equalization is introduced.

Blind equalization has been a research subject for the last few decades. The aim is to equalize the channel (to find the values of the equalizer coefficients which will invert the channel) without any apriori knowledge of the channel and training data. Several approaches have been proposed to achieve this goal. Among them, we can list (Explicit) High-Order Statistics(HOS) Methods (e.g., [1, 2, 3]), Constant Modulus Type Algorithms (e.g. [4, 5]), Subspace Methods for Multi-Branch Channels (e.g., [6, 7]), Decision-Directed and Finite Alphabet Methods (e.g., [8, 9, 10, 11]) and Maximum Likelihood Methods (e.g., [12]). This

kind of equalization is very attractive since sending training data consumes precious bandwidth. Moreover we may not have the opportunity to send training data in all of the applications even if we have enough bandwidth to spare.

For the evaluation and the comparison of these blind methods, various properties of the algorithms are to be taken into consideration. One criteria is complexity. The computational power is limited hence for an algorithm to be practically useful, its complexity must be reasonably low. As the complexity decreases, the algorithms implementation cost decreases making it very desirable for the real time applications. For example the Godard algorithm [4], one of the most popular algorithms for blind equalization, is chosen for several real time applications because of its very low complexity and simple update rule.

Another criterion for blind algorithms is their convergence behavior. The problems in convergence can be divided into two main groups as follows:

- *Ill-convergence:* This is the case when the algorithm converges to a point other then the desired one, hence removing an insufficient amount of ISI. The main reason for this kind of misbehavior is the existence of several local minima corresponding to undesired points. This problem greatly reduces the reliability of the algorithm because one can not guarantee whether the algorithm will work or not.

- *Slow-convergence:* This is the case for which the algorithm converges very slowly. The slowness is mainly caused by the attraction of the saddle points on the surface of the cost function(due to capture by saddle points [5, 13]). This problem makes the algorithm spend a large amount of data before converging to a desirable point. The slow convergence makes the algorithm unusable for time-varying channels since fast training and tracking can not be achieved.

Vembu et. al. [14] address these problems and proposes the use of the convex cost functions. As a convex cost function using the l_∞ norm of the output under a linear constraint over the equalizer coefficients is suggested. However since l_∞ norm is a non-differentiable function, its use is approximated by l_p norm with a high p value. For the exact minimization of l_∞ norm, the use of linear programming is proposed. However the high complexity of linear programming makes these l_p based algorithms not suitable for real time implementation.

The focus of this thesis is the development of low complexity algorithms that minimize the

l_∞ cost function. Our approach is based on subgradient methods for non-differentiable cost functions: although the l_∞ cost function is non-differentiable, the subgradient approach can provide us iterative algorithms with simple update rules. In addition, we can benefit from the step size selection rules, which have long been investigated in the area of subgradient optimization of convex functions, to develop algorithms with fast convergence. We show that the use of subgradient methods with proper step size selection rules leads to iterative, low complexity blind algorithms with desirable convergence behaviors.
In the next section we define the blind equalization setup that will be discussed throughout the thesis.

1.1 Blind Equalization Setup

The symbol spaced blind equalization setup that will be discussed throughout the thesis is shown in figure (1.1).

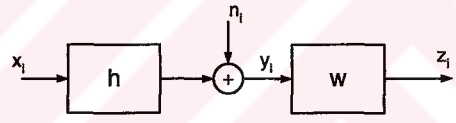

Figure 1.1: Equalization Setup.

In 1.1 :

- The input is represented by x_i which can belong to any complex or real constellation. For example $\{x_i \in \{-2 \cdot M + 1, ..., 2 \cdot M - 1\}\}$,
- The effect of the channel is represented by a linear filter $\mathbf{h}$ having coefficients $\{h_i; i \in \{0, ..., N_h - 1\}\}$,
- The noise component is represented by n_i which is assumed to be white, zero mean, Gaussian in most of the applications,
- The received signal is represented by y_i,

- The equalizer coefficients are represented by $\{w_i; i = 0, ..., N_w - 1\}$. The equalizer vector can be defined as $\mathbf{w} = [w_1 \cdots w_n]$. Equalization process is based on the search for the correct elements of this vector,

- The output of the equalizer is represented by z_i.

In this thesis we will also deal with fractionally spaced equalizers which are implemented in systems with sampling rates different than symbol rates. In figure (1.2) the equalization setup for fractionally spaced equalizers is shown. In this setup the input is oversampled by a factor of p and the equalizer can be considered to have a vector with input dimension p. In other words the output of this equalizer is a linear combination of the outputs of the p subequalizers (by subequalizer we mean the portion of the equalizer corresponding one one phase of the oversampled sequence). In figure (1.2) we can see that the fractionally spaced equalizer is no different then an equalizer for SIMO channels hence from this point it will be treated as one. In chapters 3.3 and 4 detailed information about fractionally spaced

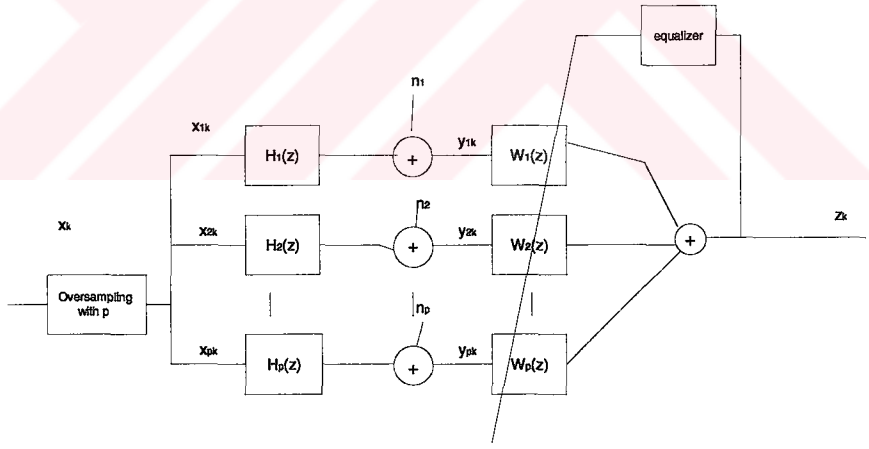

Figure 1.2: Fractionally Spaced Equalizer.

equalization is given.

For all of the blind equalization approaches that we consider in the thesis, some common assumptions about the equalization setup are made. These assumptions can be listed as follows:

- The channel is assumed to be linear and time-invariant. Since signal is distorted by the linear effects such as multipath, this is a generally accepted and reasonable assumption for the channel,

- x_i can be an n-dimensional PAM or QAM signal. For example $\{x_i \in \{-2 \cdot M+1, ..., 2 \cdot M-1\}\}$,

- x_i is assumed to be iid,

1.2 Problem Definition

The problem of blind equalization can be defined as finding $\mathbf{w}$ that eliminates the effect of $\mathbf{h}$ on $\mathbf{x}$. In other words, the equalization problem is to find the equalizer coefficients that will make the equalizers impulse response equal to the inverse of the channels impulse response. The correct $\mathbf{w}$ will satisfy the condition below:

$$\mathbf{h} * \mathbf{w} = \delta_{n-d} \cdot c. \tag{1.1}$$

In (1.1), d is an arbitrary delay and c is a non-zero complex constant. Since there is no information about the delay and the magnitude element, all of the $\mathbf{w}$ satisfying (1.1) will be a solution for the problem. In other words $\mathbf{w}$ is a shifted and scaled convolutional inverse of the channel.

In supervised and adaptive blind equalization it is generally not possible to obtain the exact inverse of the channel especially under the FIR constraint on equalizer coefficients. Therefore, the adaptive equalization is posed as an optimization task where the equalizer is to be chosen as "close" as possible to the inverse of the channel. Of course the term "closeness" depends on the distance measures used. Throughout the thesis we'll use the following two closeness measures,(which actually measures the closeness of the combined channel and equalizer to a delta function);

- *Open eye measure(Normalized peak distortion)*: $\rho(c) = \frac{\sum |c_i| - p}{p}$,

- *ISI (Normalized inter symbol interference energy)*: $ISI(c) = \frac{\sum |c_i|^2 - p^2}{p^2}$,

where $p = \max_i |c_i|$. Both measures are nonnegative and get zero value only under the perfect equalization condition.

So far the concept of blind equalization is explained. The setup for blind equalization is shown and some common assumptions for this setup are given. As stated before there is no information about the impulse response of the channel. Furthermore, there is no training signal available that is sent as the input.So the output of the channel with the assumptions on the input are all the information we have.

1.3 Thesis Overview

The organization of the thesis is as follows: Information on relevant mathematical background on convexity and subgradients are given in Chapter 2. In Chapter 3 the existing approaches for the blind equalization problem are summarized. This chapter is followed by chapter 3.3 which gives insight about the cost functions. Chapter 4 is the main chapter where subgradient optimization based algorithms that we developed are introduced. Chapter 5 provides information on the complexity of the algorithms. Chapter 6 contains the simulation results of the proposed and previously developed algorithms. Finally conclusions are given in Chapter 7.

Chapter 2

MATHEMATICAL BACKGROUND

The blind equalization approach that we proposed in this thesis is based on convex optimization methods in particular subgradient techniques. Basic motivation for the use of convex optimization is the unimodel structure of the convex cost functions use of which avoids ill-convergence and slow convergence problems encountered in adaptive algorithms with multi-modal cost surfaces.

The non-differentiability of the convex cost functions proposed for blind equalization is a major obstacle in obtaining low complexity algorithms, as the gradient search is the most popular approach for developing such algorithms. Fortunately, the subgradient approach will enable us to develop low complexity algorithms suitable for real time implementations. This chapter aims to provide basic mathematical background covering convex sets, functions and subgradient techniques.

2.1 Convexity

In this section definitions of "convex set" and "convex function" will be made, then the properties of a convex set will be given. Detailed information on convexity can be found in [15, 16].

2.1.1 Convex Set

A convex set can be defined as:

$C \subset \Re^n$ *is called convex if*

$$(\alpha x_1 + (1-\alpha)x_2) \in C, \qquad \forall x_1, x2 \in C, \quad \forall \alpha \in [0,1]. \tag{2.1}$$

Equation 2.1 states that if two points are elements of a convex set, then all the points belong to the straight line section between these two points also belong to this convex set.

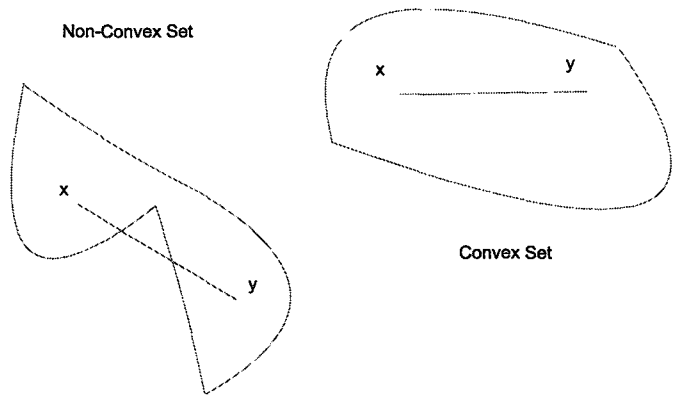

Figure 2.1: Convex and Non-convex Sets.

Some important propositions about convex sets are as follows:

- The intersection of convex sets is also a convex set.
- Let C_1 and C_2 be convex sets, then $C_1 + C_2$ is also a convex set.
- For two positive scalars λ_1,λ_2 and a convex set C, $(\lambda_1 + \lambda_2)C = \lambda_1 C + \lambda_2 C$.
- Closure and interior of a convex set is convex.
- An affine image (or inverse image) of a convex set is convex.

Convex-Hull of a set A is the smallest convex set containing all of the elements of A.

2.1.2 Convex Function

A convex function is defined as follows:
Let C be a convex set and $C \subset \Re^n$. f is convex if

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y), \qquad \forall x, y \in C, \quad \forall \alpha \in [0,1]. \tag{2.2}$$

In figure (2.2) a simple convex function is shown.

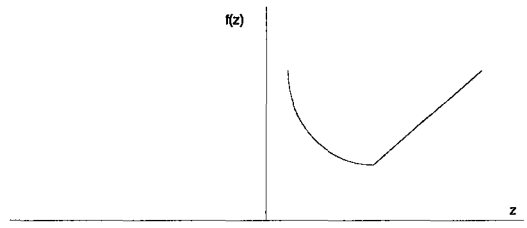

Figure 2.2: A convex function

2.2 Subgradient

Let $f : \Re^n \rightarrow \Re$ be a convex function. A vector $\mathbf{d} \in \Re^n$ is a subgradient of f at point $x \in \Re$ if

$$f(z) \geq f(x) + (z - x)'\mathbf{d}, \qquad \forall z \in \Re. \tag{2.3}$$

If f is concave then $\mathbf{d}$ is a subgradient of f if $-\mathbf{d}$ is a subgradient of $-f$(a convex function). The set of subgradients of f at point x is called the subdifferential of f at x. Subdifferential of f is denoted by ∂f.

$$\forall \mathbf{d} \in \partial f(x) \qquad f(z) \geq f(x) + (z - x)'\mathbf{d}, \quad \forall z \in \Re. \tag{2.4}$$

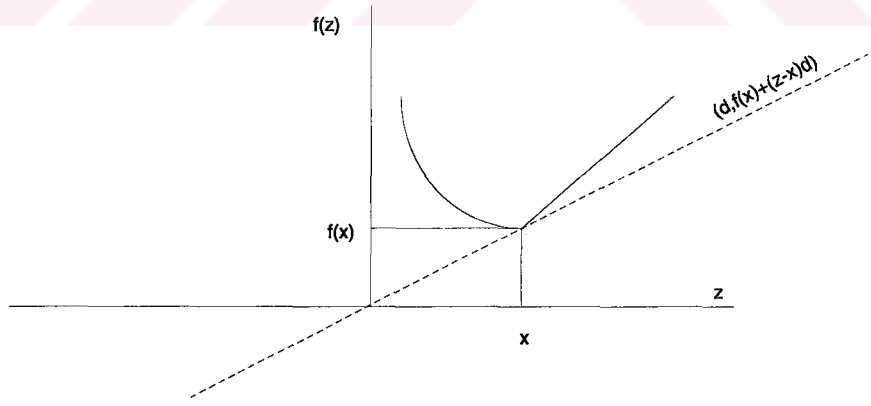

Figure 2.3: The tangential plane formed by a subgradient at point x.

In Figure (2.3) one can see that the tangential plane formed by a subgradient is always a lower bound (or upper bound if concave) for that function. Moreover at point x function f

is not differentiable but subgradients exists. The existence of a subgradient is independent of differentiability (which is one of its most important properties). Detailed information about subgradients can be found in [15, 16, 17].

Now that subgradient is defined, it is time to introduce the general subgradient algorithm. In the next section first a brief information on the gradient-descent methods is given, then the subgradient algorithm is introduced.

2.3 The Subgradient Algorithm

The gradient-descent methods use the first order derivative information to update the search vector such that the objective function value is iteratively improved. The goal is to eventually reach the neighborhood of a global or a local optimal point, depending on the property of the cost function.

Given $f(\mathbf{w})$ defined over set S is a differentiable cost function to be minimized, the gradient descent method consists of iterations of the form

$$\mathbf{w}^{(i+1)} = \mathcal{P}_S \left\{ \mathbf{w}^{(i)} - \mu^{(i)} \nabla f(\mathbf{w}^{(i)}) \right\}, \tag{2.5}$$

where $\mathcal{P}_S$ is the projection on set S and the step size μ is chosen small enough to ensure the decrease in the cost function.

Various adaptive and blind algorithms have been based on this structure due to its low complexity [18, 19]. The most famous examples are the Least Mean Square (LMS) algorithm in supervised adaptive filtering and the Constant Modulus Algorithm (CMA) in blind equalization.

In the CMA case, for the equalization setup in Figure 1.1, the cost function is given by [5]

$$f(\mathbf{w}) = E((\gamma - |z|^2)^2), \tag{2.6}$$

γ is the level corresponding to the constant modulus. The update equation corresponding to the CMA cost function in (2.6) is given by

$$\mathbf{w}^{(i+1)} = \mathbf{w}^{(i)} + \mu^{(i)} \underbrace{\mathbf{y}_i z_i^* (\gamma - |z_i|^2)}_{\hat{\nabla} f(\mathbf{w}^{(i)})}, \tag{2.7}$$

where the stochastic gradient is replaced by its estimate. Note that the projection operator in (2.5) doesn't appear in (2.7). The reason is that the CMA cost function in (2.6) is defined over the whole $\mathcal{R}^{N_w}$ and therefore the projection operator is the identity operator.

The gradient-descent methods are applicable to only differentiable functions. The non-differentiable counterpart of the gradient-descent algorithm is the subgradient projection method in which the gradient vector is simply replaced by a subgradient vector:

$$\mathbf{w}^{(i+1)} = \mathcal{P}_S \left\{ \mathbf{w}^{(i)} - \mu^{(i)} \mathbf{g}^{(i)} \right\}, \tag{2.8}$$

where $\mathbf{g}^{(i)}$ is a subgradient picked from the subdifferential set $\partial f(\mathbf{w}^{(i)})$. Although the subgradient algorithm looks very much like the gradient descent algorithm, in the subgradient iteration it may happen that

$$f(\mathbf{w}^{(i+1)}) > f(\mathbf{w}^{(i)}) \tag{2.9}$$

for any $\mu^{(i)} > 0$ [15]. However, if the $\mu^{(i)}$ parameter sequence is properly chosen then $\mathbf{w}^{(i)}$ can be made to converge to the optimal point $\mathbf{w}^*$. In fact the selection of the step size is a crucial point and it has been a research focus for several decades where various step size selection schemes have been developed. Before introducing some of the previously proposed step sizes, the sufficient conditions for the convergence of the algorithm is given in ZLDS step size rule.

2.3.1 Zero-Limit-Divergent-Sum (ZLDS) Step Size Rule

One major result about the selection of the step size parameter $\mu^{(i)}$ is due to Polyak ([20, 21]): if the following two conditions hold

$$\lim_{i \to \infty} \frac{\mu^{(i)}}{||\mathbf{g}^{(i)}||_2} = 0, \tag{2.10}$$

$$\sum_{i=0}^{\infty} \frac{\mu^{(i)}}{||\mathbf{g}^{(i)}||_2} = \infty, \tag{2.11}$$

then $\lim_{i \to \infty} \mathbf{w}^{(i)} = \mathbf{w}^*$. Therefore, Equation 2.10 (Zero Limit) and Equation 2.11 (Divergent Sum) provide sufficient conditions for convergence based on the step size selection.

A simple example of a step size satisfying ZLDS rule is

$$\mu^{(i)} = \frac{\mu_0}{i+1}, \tag{2.12}$$

which satisfies the conditions in Equations 2.10-2.11. This step size rule has been studied in [22]. Under certain assumptions regarding the cost surface, it is shown that the square distance to the optimal point $\mathbf{w}^*$ decreases with a sublinear convergence behavior, i.e. we can approximately write

$$\|\mathbf{w}^* - \mathbf{w}^{(i)}\|_2^2 \leq \alpha \|\mathbf{w}^* - \mathbf{w}^{(0)}\|_2^2 \tag{2.13}$$

where $\alpha \propto \frac{1}{k^p}$ for some parameter p which is dependent on the cost surface and μ_0.

2.3.2 Relaxation Step Size Rule

A dynamic step size rule, known as relaxation rule[15, 23], is given by

$$\mu^{(i)} = \gamma^{(i)} \frac{(f(\mathbf{w}^{(i)}) - f^*)}{||\mathbf{g}^{(i)}||_2^2}, \tag{2.14}$$

where

$$0 < \gamma^l \leq \gamma^{(i)} \leq \gamma^u < 2, \tag{2.15}$$

and f^* is the minimum value of $f(\mathbf{w})$. If the step size is selected based on this rule then it is guaranteed that

$$||\mathbf{w}^{(i+1)} - \mathbf{w}^*||_2 < ||\mathbf{w}^{(i)} - \mathbf{w}^*||_2 \qquad \forall i \tag{2.16}$$

i.e., the distance to the optimal vector decreases monotonically[22]. The proof of this monotonically convergent behavior is as follows:

$$\|\mathbf{w}^{(i+1)} - \mathbf{w}^*\|^2 = \|(\mathcal{P}_S \left\{\mathbf{w}^{(i)} - \mu^{(i)}\mathbf{g}^{(i)}\right\}) - \mathbf{w}^*\|^2 \tag{2.17}$$

$$= \|\mathbf{w}^{(i)} - \mu^{(i)}\mathbf{g}^{(i)} - \mathbf{w}^*\|^2 \tag{2.18}$$

$$= \|\mathbf{w}^{(i)} - \mathbf{w}^*\|^2 - 2(\mathbf{w}^{(i)} - \mathbf{w}^*)'\mathbf{g}^{(i)}\mu^{(i)} + \|\mu^{(i)}\mathbf{g}^{(i)}\|^2 \tag{2.19}$$

$$\leq \|\mathbf{w}^{(i)} - \mathbf{w}^*\|^2 - 2(f(\mathbf{w}^{(i)}) - f^*)\mu^{(i)} + \|\mu^{(i)}\mathbf{g}^{(i)}\|^2 \tag{2.20}$$

$$= \|\mathbf{w}^{(i)} - \mathbf{w}^*\|^2 - 2\gamma^{(i)}\frac{(f(\mathbf{w}^{(i)}) - f^*)^2}{||\mathbf{g}^{(i)}||_2^2} + (\gamma^{(i)})^2\frac{(f(\mathbf{w}^{(i)}) - f^*)^2}{||\mathbf{g}^{(i)}||_2^2} \tag{2.21}$$

$$= \|\mathbf{w}^{(i)} - \mathbf{w}^*\|^2 - (2 - \gamma^{(i)})\gamma^{(i)}\frac{(f(\mathbf{w}^{(i)}) - f^*)^2}{||\mathbf{g}^{(i)}||_2^2} \tag{2.22}$$

When $(2-\gamma^{(i)})\gamma^{(i)}\frac{(f(\mathbf{w}^{(i)})-f^*)^2}{||\mathbf{g}^{(i)}||_2^2} < 0$, (2.16) is guaranteed and this inequality is satisfied only if $0 < \gamma^l \leq \gamma^{(i)} \leq \gamma^u < 2$ (The projection operator is eliminated in (2.17) since the convex

set is the subspace itself and the inequality in (2.19) is a direct result of the definition of subgradient).

The choice of μ as in Equation 2.14 also provides a faster rate of convergence (than the step size in Equation 2.12) . The convergence rate of this scheme has been studied in several references (see for example [24, 22, 23]). For this step size selection, a bound for the convergence rate can be derived[22] based on the assumption that, there exists some κ for which

$$f(\mathbf{w}) - f^* \geq \kappa \|\mathbf{w}^* - \mathbf{w}\|_2 \qquad \forall \mathbf{w} \in S. \tag{2.23}$$

Using this assumption, and assuming that $\|\mathbf{g}^{(i)}\|_2 \leq C$, it can be shown that[22]

$$\|\mathbf{w}^* - \mathbf{w}^{(i)}\|_2 \leq q^i \|\mathbf{w}^* - \mathbf{w}^{(0)}\|_2, \tag{2.24}$$

where

$$q = \sqrt{1 - \gamma^l(2 - \gamma^u)\frac{\kappa^2}{C^2}}. \tag{2.25}$$

For piecewise linear functions of the form $f(\mathbf{w}) = \|A\mathbf{w} + \mathbf{b}\|_\infty$, Brannlund shows that [23] κ in (2.23) can be computed as the solution of the optimization problem

$$\begin{aligned} &\min && z \\ &\text{s.t.} && AA^T\alpha \leq z\mathbf{1} \\ &&& \alpha^T AA^T \alpha = 1 \\ &&& \alpha \geq \mathbf{0} \end{aligned} \tag{2.26}$$

Unfortunately, q factor obtained based on Equation 2.25 provides only a pessimistic upper-bound for the rate of convergence.

2.3.3 Relaxation Step Size Rule With Variable Target Values

The relaxation step size rule in Equation 2.14 requires a priori knowledge of the optimal value of the function to be optimized which is not reasonable in most cases. The use of an estimate of f^*, instead of f^* have been investigated in several references (see for example [25, 26, 27]) such that the subgradient update rule takes the form

$$\mathbf{w}^{(i+1)} = \mathcal{P}_S \left\{ \mathbf{w}^{(i)} - \alpha^{(i)} \frac{(f(\mathbf{w}^{(i)}) - \hat{f}^*)}{\|\mathbf{g}^{(i)}\|_2^2} \mathbf{g}^{(i)} \right\}. \tag{2.27}$$

Bazaraa and Sherali had a pioneering work on algorithms which don't require the knowledge of f^*[28]. Along the same lines, Kim et. al. proposed a variable target method where $\hat{f}_*^{(i)}$ is updated as a part of the algorithm [29].

Recently Goffin and Kiwiel proposed a simple subgradient method[30] which is relatively low complexity and which is guaranteed to converge. The algorithm is a variable target method which is based on the following two simple facts[15]:

1. if $\hat{f}^*$ in (2.27), is an overestimate of the optimal value f^*, then either

   - for some k, $f(\mathbf{w}^{(k)}) \leq \hat{f}^*$ or,
   - $\mathbf{w}^{(i)}$ converges to an $\bar{\mathbf{w}}$ such that $f(\bar{\mathbf{w}}) \leq \hat{f}^*$,

   such that overestimate of f^* is detected,

2. if $\hat{f}^*$ in (2.27) is an underestimate of the optimal value f^*, then it can be shown that

   $$\sum_{i=0}^{\infty} \frac{f(\mathbf{w}^{(i)}) - \hat{f}^*}{||\mathbf{g}^{(i)}||_2} = \infty \tag{2.28}$$

   so that one can detect underestimate condition by checking whether the finite summation

   $$\sum_{i=0}^{K} \frac{f(\mathbf{w}^{(i)}) - \hat{f}^*}{||\mathbf{g}^{(i)}||_2} \tag{2.29}$$

   is above some threshold parameter.

Therefore, one can start with an arbitrary $\hat{f}^*$ and later update it based on the conditions above.

Nedic and Bertsekas proposed [22] a simpler adaptive step size algorithm with unknown f^* where the estimate of the optimal value is given by

$$\hat{f}_*^{(i)} = \min_{0 \leq j \leq i} f(\mathbf{w}^{(i)}) - \delta^{(i)}. \tag{2.30}$$

Here the $\delta^{(i)}$ is updated according to

$$\delta^{(i+1)} = \begin{cases} \sigma\delta^{(i)} & \text{if } f(\mathbf{w}^{(i)}) < \hat{f}_*^{(i)}, \\ \max\{\beta\delta^{(i)}, \delta\} & \text{if } f(\mathbf{w}^{(i)}) \geq \hat{f}_*^{(i)}, \end{cases} \tag{2.31}$$

where $\delta^{(0)}, \delta, \sigma, \beta$ are fixed positive constants with $\beta < 1$ and $\sigma \geq 1$.

Sherali et. al. [31] also proposed an algorithm with the a similar complexity to the Goffin's algorithm outlined above. The algorithm proposed in [31] also incorporates the so called dilation technique to increase the speed of convergence.

Information on variable target value concludes the mathematical background chapter. In the next chapter brief information about the previous work on blind equalization will be given.

Chapter 3

PREVIOUS WORK

3.1 Classification Of Blind Equalization Methods

The algorithms developed so far can be classified in three main groups.

- Subspace Methods,
- Higher Order Statistics Methods,
- Maximum Likelihood Methods.

In this chapter, brief information about each method will be given and some of the popular algorithms will be introduced. A more detailed framework can be found in [12].

3.1.1 Subspace Methods

For SISO (single input single output) channels equalization process can be made only by using higher-order statistics(higher then second) because of the fact that phase information about the channel can not be obtained by using second order statistics of the channel . Most of the algorithms using higher order statistics has drawbacks like slow, ill-convergence, the need for unreasonably high data lengths etc.

To equalize the channel using only the second order statistics is possible only if the channel has multiple outputs. The work of Kaliath, Tong and Xu [32] has shown how to identify a channel using only second order statistics. The idea was to form a multiple output channel by using an antenna array combined with fractional sampling.

One of the most popular algorithms using subspace methods is the MUSIC algorithm[33]. MUSIC detects frequencies in a signal by performing eigenvalue decomposition on the covariance matrix of a data vector from the samples of the received signal. The algorithm assumes a structure about the channel and uses the fact that signal vectors are orthogonal to noise subspace. By using the orthogonality the algorithm reaches a scaled version of the

channel impulse response. ESPRIT [34] is another important equalization algorithm using subspace method.

The subspace methods have high convergence speeds and reliability. However as we can see these methods can only be used for channels having multiple outputs. Also the algorithms using these methods have high complexity which is an important drawback for this class of algorithms. Finally since the algorithm assumes a structure it fails when the assumption is incorrect.

3.1.2 Higher Order Statistics Methods

The subspace methods can be used only in channels having multiple outputs. Most of the time we are faced with SISO channels which make subspace methods unusable. To equalize a SISO channel we need more then the second order statistics of the channel hence we make use of higher order statistics of the channel and this is what HOS methods are all about. [4, 35, 36] have proposed some of the popular equalization algorithms using HOS methods. HOS methods can be divided into two main groups:

- *Explicit HOS* : These type of algorithms involved in the use of the higher order statistics of the channel directly. [35, 36] are good examples for this type of algorithms since they are using kurtosis function.

- *Implicit HOS* : These type of algorithms do not use the higher order statistics directly, however the result leads to the use of higher order statistics of the channel. [4] is a good and famous example for this case.

3.1.3 Maximum Likelihood Methods

Maximum Likelihood (ML) is an important method in statistics with the property that under ideal conditions variance of ML estimator converges to a lower bound(Cramer-bound) for all unbiased estimators. Because of this property it has been applied in many fields of signal processing [37] especially in sequence estimation and decoding problems in digital communications[38]. For the same reason various ML based algorithms have been proposed in blind equalization where we can classify them under two main categories:

- *Deterministic Maximum Likelihood*: In this case the input sequence is treated as an unknown deterministic sequence[12].

- *Stochastic ML*: In this case the input sequence is considered as a random process.

In both approaches the equalizer coefficients or the channel coefficients are obtained through the optimization of the likelihood function.

$$\hat{x} = arg_{max\ x}\ p(y \mid x) \tag{3.1}$$

It is in general not possible to obtain a closed form solution to this problem. Therefore the procedure for obtaining the solution for the ML cost function is generally gradient search based.
As the likelyhood cost surface is multimodal in general ML based blind algorithms reported to suffer from ill and slow convergence[12].

3.2 Existing Popular Algorithms Using Non-Convex Cost Functions

3.2.1 CMA

The constant modulus algorithm (CMA) probably the most well known and the most popular approach in blind adaptive equalization. CMA is an iterative blind algorithm with a simple update rule hence it has easy implementation. For some constellations magnitude of the transmitted signal is constant (2-PAM, 4-QAM etc.). This information is used to form the cost function of the CMA. This cost function is

$$J(\mathbf{w}) = E(||y_i|^2 - c|^2), \tag{3.2}$$

where c is some positive constant value. The condition that minimizes this cost function is $|y_i|^2$ being equal to c. In other words the magnitude of the output of the equalizer is constant and equal to c. The value of c is chosen without any criteria(it may be chosen based on the power level of input to increase the convergence speed) since each of them will lead algorithm to a scaled version of the optimum equalizer point($\mathbf{w}^*$). For this cost function the error is defined as,

$$e = |y_i|^2 - c. \tag{3.3}$$

The direction vector for each step of the algorithm is represented by $\mathbf{g}^{(k)}$ which is defined as,

$$\mathbf{g}^{(k)} = [\ x_{m+k} \quad \dots \quad x_{1+k}\ ]^T. \tag{3.4}$$

Vector $\mathbf{g}^{(k)}$ is the convolution vector for the k^{th} iteration. Now the update rule can be written as:

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \mu^{(k)} \cdot \mathbf{g}^{(k)} \cdot e, \tag{3.5}$$

where $\mu^{(k)}$ is an adaptive coefficient determining the step size. In each iteration of the algorithm, output vector $\mathbf{y}$ is calculated. By using $\mathbf{y}$ the error is calculated. Finally the equalizer vector $\mathbf{w}$ is updated as in (3.5)(There are modified versions of CMA and their update rules are shown in table (3.1)). The algorithm continues to run until an acceptable ISI value is reached.

As stated before the CMA greatly suffers from ill and slow convergence. This is because of the non-convex structure of its cost function ($J(\mathbf{w})$). As these local minima are the stationary points where gradient is equal to the all-zeros vector, a gradient search based algorithm can converge at them. When one of these points are reached, $\mathbf{g}_k$ becomes zero and $\mathbf{w}$ stops changing. When this is the case there is nothing to be done except to restart the algorithm and hope that it converges to the global minimum next time. Also saddle points greatly reduces the speed of convergence when the equalizer vector passes through their neighborhood.

In table (3.1) se-CMA step size uses $sgn(e)$ instead of e. NCMA is a normalized version of CMA using only the direction information of the vector $\mathbf{g}$. In the step size of NWCMA, $\mathbf{G}$ denotes the window matrix. Each row of $\mathbf{G}$ is another convolution vector for CMA, and instead of one now several vectors are used to estimate the direction of the algorithm. More versions of CMA algorithm is present today.

As seen so far CMA has a very simple update rule and low complexity making the algorithm suitable for hardware only implementation. The algorithm has two drawbacks. The convergence speed is very slow due to the update rule ,the search direction and non-convex structure of the cost function. Cost function's non-convex structure also causes ill-convergence problem. Since there are false minima the algorithm can easily converge to these points. This causes insufficient removal of ISI(3.1).

Name	Step Size
CMA	$\lambda \cdot \mathbf{g} \cdot e$
SE-CMA	$\lambda \cdot \mathbf{g} \cdot sgn(e)$
NCMA	$\lambda \cdot \mathbf{g} \cdot (\mathbf{g} \cdot \mathbf{g}^T)^{-1} \cdot e$
NWCMA	$\lambda \cdot \mathbf{G} \cdot (\mathbf{G} \cdot \mathbf{G}^T)^{-1} \cdot e$

Table 3.1: Different Step Sizes For CMA

The complexity of the CMA is very low. In each step there is a vector multiplication to

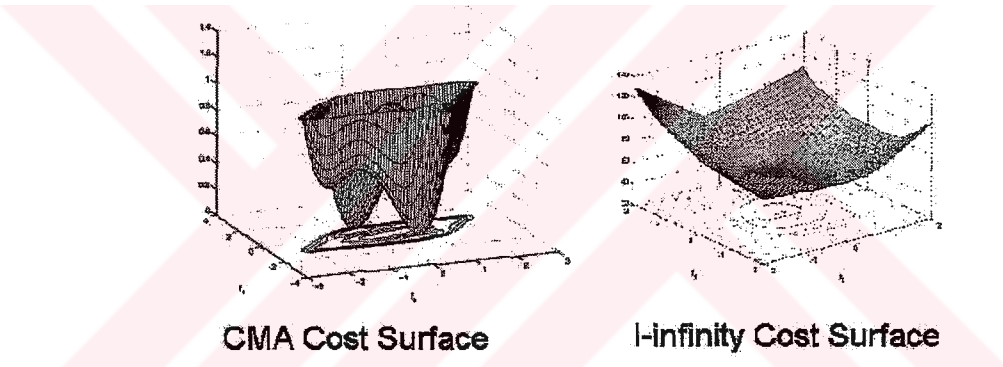

Figure 3.1: Convex and non-convex cost surfaces.

calculate the current output followed by simple summations to calculate the new equalizer vector. Hence the complexity of CMA can be given as N_w additions and multiplications for each step.

The easy implementation and very low computational complexity are the two main reasons for CMA being the most popular blind equalization algorithm. Understanding this algorithm gives important information about blind algorithms.

The CMA uses implicit HOS hence can be used in symbol spaced channels.

3.2.2 Shalvi Weinstein

The algorithm is introduced by Ofir Shalvi and Ehud Weinstein [35]. The suggested equalization criterion is,

$$maximize \quad |K(z)| \quad while \quad E|z|^2 = E|a|^2. \tag{3.6}$$

In equation 3.6 K is the kurtosis function function which is defined as:

$$K(z) = E|z|^4 - 2E^2|z|^2 - |Ez^2|^2. \tag{3.7}$$

The update rule for the algorithm is given as :

$$\mathbf{w}' = \mathbf{w} + \delta \cdot \nabla_w \phi. \tag{3.8}$$

In equation 3.8 $\nabla_w \phi$ is the gradient,δ is the step size and $\mathbf{w}$ is the equalizer vector.

$$\phi = sgnK(x)[E|z|^4 - |Ez^2|^2 + \gamma_1 E^2|z|^2 + 2\gamma_2 E|z|^2] \tag{3.9}$$

where

$$\gamma_1 = -2 - \frac{(1+\alpha)K(a)}{E^2|x|^2} \tag{3.10}$$

and

$$\gamma_2 = \alpha \frac{K(x)}{E|a|^2}. \tag{3.11}$$

Up to this point we can see that the algorithm makes explicit use of HOS (kurtosis function). This enables the applicability of the algorithm to symbol spaced channels, however it also increases the complexity of the algorithm, making it very inappropriate for hardware only implementation. The algorithm is nearly unaffected from the distribution of the input except the case where the input has Gaussian distribution, making the kurtosis function zero. The noise performance is very good when the noise is i.i.d. Gaussian because the cost functions noise component is zero for this case.

3.2.3 Super Exponential Method

Another method proposed by Ofir Shalvi and Ehud Weinstein is the Super Exponential Method [36]. This time instead of linear prediction, a nonlinear recursive method is proposed.

The update rule of the algorithm can be written as:

$$\mathbf{w}' = \mathbf{R}^{-1} \cdot \mathbf{d}, \tag{3.12}$$

and

$$\mathbf{w}'' = \frac{1}{\sqrt{(\mathbf{w}') + \mathbf{R}\mathbf{w}}}\mathbf{w}'. \tag{3.13}$$

In equation 3.12 $\mathbf{R}$ and $\mathbf{d}$ are defined as:

$$R_{nm} = \frac{\widehat{cum}(y_{t-m}; y^*_{t-n})}{cum(x_t; x^*_t)}, \tag{3.14}$$

$$d_n = \frac{\widehat{cum}(z_t : p; z^*_t : q; y^*_{t-n})}{cum(x_t : p; a^* : q+1)}. \tag{3.15}$$

Above *cum* is the cumulative function defined as:

$$cum(y_{t-m}y^*_{t-n}), \tag{3.16}$$

and

$$cum(z_t : p; z^*_t : q; y^*_{t-n})|_{p=q=1} = E(|z_t|^2 y^*_{t-n}), \tag{3.17}$$

$$cum(z_t : p; z^*_t : q; y^*_{t-n})|_{p=2,q=1} = E(|z_t|^2 z_t y^*_{t-n}) - 2E(|z_t|^2)E(z_t y^*_{t-n}) - E(z_t^2)E(z^*_t y^*_{t-n}). \tag{3.18}$$

$\widehat{cum}$ is an estimate of *cum*. As seen from the update rule, the current value of the equalizer does not depend linearly on the previous equalizer coefficients. In each step the recursion above is repeated until a sufficient value for ISI is reached.

The algorithm has very high convergence speed (the decay in ISI is nearly super exponential). Also like the previous algorithm the only assumption is the input having a non-Gaussian distribution. However because of the need for calculating the power of some variables, the algorithm suffers greatly from finite precision effects which makes it hard for hardware implementation. Another drawback is ill convergence. Cost function is not convex hence the algorithm can converge to an undesirable point. This greatly reduces the algorithms reliability.

3.3 Blind Equalization By Using Convex Cost Functions

Most of the blind algorithms are based on the minimization of some cost function. The characteristics of these algorithms (such as convergence rate, reliability) strongly depend on the cost function used for the algorithm. When there are local minima on the cost surfaces of these type of algorithms, these points can be the target of the algorithms causing ill-convergence. Another problem is caused by the saddle points, where the algorithms speed greatly reduces when it passes through a neighborhood of any of these saddle points(slow-convergence problem). Convex cost functions don't have these problems since there are no saddle points nor local minimas on their surfaces. These properties makes convex cost functions very desirable for blind equalization.

3.3.1 Necessary Conditions For Convex Cost Functions

When choosing cost functions, one has to consider several criteria. For a convex function to be a cost function of any blind equalization method, first of all the points that equalize the channel must be the global minimum for that function. Convexity assures us that there are no local minima, however the point(s) (Here the cost function is defined on the equalizer coefficients $\mathbf{w}$, hence a points coordinates correspond to the equalizer coefficient values) minimizing this function may not lead to the optimum equalizer coefficients. One common example is the convergence to the all zero setup. For example when we consider the infinity norm of the output, clearly the all zero setup minimizes this convex function. To avoid this problem, a linear constraint is introduced. Since the constraint is linear, the resulting function is still convex. Moreover this constrained function has its global minimum on the points equalizing the channel.

Another important criteria is being delay insensitive[14]. Since there is no information about the delay in blind equalization, the cost function must be independent of delay. In other words when the algorithm starts equalizing, the characteristic of the resulting cost function must be the same all the time. This property is very crucial for the reliability of the algorithm.

3.3.2 l_∞ Norm As A Convex Cost Function

Due to the local optima and saddle point structure of their cost functions, the CMA and some other blind equalization algorithms suffer from the slow and ill convergence problems. This problem caused the search for the cost functions with better geometrical structure. The reference [14] investigates the properties of convex functions as the candidates for being cost functions in blind equalization. In the same reference, minimizing the l_∞ norm of the output of the equalizer, under a linear constraint over equalizer coefficients, is proposed as a blind equalization approach. Considering the equalization setup in Figure 1.1, under the assumptions that

- the input constellation has the maximum magnitude symmetry around zero such that $\max x_k = (2 \cdot M - 1)$ and $\min x_k = -2 \cdot M + 1$, and,
- $\{x_k\}$ is sufficiently rich in terms of variations in time,

it can be shown that [14]

$$||z||_\infty = (2 \cdot M - 1)||c||_1 \tag{3.19}$$

where $c_i = h_i * w_i$ is the combined impulse response of the channel and the equalizer. Therefore, minimizing the l_∞ norm of the output is equivalent to minimize the l_1 norm of the overall impulse response. A linear constraint of the form

$$\mathbf{d}^T\mathbf{w} = e \qquad \mathbf{d} \neq 0, e \neq 0 \tag{3.20}$$

is needed to avoid the all-zero solution($\mathbf{w} = \mathbf{0}$).

The equivalence between the minimization of the l_1 norm of $\{c\}$ and the equalization of the channel $\{h_i\}$ is given by the following proposition which is proven in [14]:

Proposition : *Given the impulse response $\{h_i; i \in \mathcal{Z}\}$ and the impulse response of its ideal convolutional inverse $\{d_i; i \in \mathcal{Z}\}$ such that*

$$d_n * h_n = \delta_n, \tag{3.21}$$

where $|d_n|$ is assumed to have a single maximum point located at m, then, the solution to the problem

$$\begin{aligned} &\textit{minimize } ||h_n * w_n||_1 && \textit{(Problem 1)} \\ &s.t.\ w_0 = 1 \end{aligned}$$

is given by

$$w_n = \frac{d_{n+m}}{d_m} \qquad \forall n. \tag{3.22}$$

The proof for this statement is as follows [14] :

Suppose $\mathbf{w}^*$ is the vector satisfying the equality in 3.21. Then any equalizer vector can be written as $\mathbf{w}' + \mathbf{w}^*$ where $\mathbf{w}^*$ is stated above and $\mathbf{w}'$ corresponds to the difference between equalizer vector and $\mathbf{w}^*$. The cost functions can be written as:

$$J(\mathbf{w}) = ||h_n * w_n||_1, \tag{3.23}$$

$$J(\mathbf{w}) = J(\mathbf{w}' + \mathbf{w}^*), \tag{3.24}$$

$$J(\mathbf{w}' + \mathbf{w}^*) = \sum_i |\mathbf{h}_i * \mathbf{w}'_i + \mathbf{h}_i * \mathbf{w}^*_i|. \tag{3.25}$$

Also we have:

$$\mathbf{w}'_0 = 0 \tag{3.26}$$

and

$$\mathbf{w}^* * \mathbf{h} = A\delta_{n-k} \tag{3.27}$$

where $A = 1/d_k$. Also $\epsilon_i = \mathbf{h}_i * \mathbf{w}'_i$.Now we can write:

$$J(\mathbf{w}) - J(\mathbf{w}^*) = \sum_i |\mathbf{h}_i * \mathbf{w}'_i + \mathbf{h}_i * \mathbf{w}^*_i| - \sum |\mathbf{h}_i * \mathbf{w}^*| \tag{3.28}$$

$$= \sum_{i \neq k} |\epsilon_i| + |\epsilon_k + A| - |A| \tag{3.29}$$

$$\geq sgn(A)[\epsilon_k + \sum_{i \neq k} \epsilon_i \mathbf{w}^*_{k-i}] = sgn(A)[\epsilon_k * \mathbf{w}^*] \tag{3.30}$$

$$= |A|\mu_0 = 0. \tag{3.31}$$

The facts for the equation above are:

$$|\epsilon_k + A| - |A| \geq sgn(A)\epsilon_k \tag{3.32}$$

$$\sum_{i \neq k} |\epsilon_i| \geq \pm \sum_{i \neq k} \epsilon_i \mathbf{w}^*_{k-i} \tag{3.33}$$

since $|\mathbf{w}^*_i \leq 1|$.

As a result, under the previous assumptions made about $\{x_i\}$, the equalizer coefficients $\{w_i; i \in \mathcal{Z}\}$ obtained by solving the convex optimization problem

$$\begin{aligned} &\text{minimize } ||z||_\infty \qquad \textit{(Problem 2)} \\ &\text{s. t. } w_L = 1 \end{aligned}$$

for some $L \in \mathcal{Z}$, will be the scaled and time shifted inverse of $\{h_i\}$ such that

$$h_n * w_n = G\delta_{n-d} \tag{3.34}$$

for some magnitude G and delay d[14]. Hence, the selection of which tap to be fixed is strictly related to the optimal delay selection problem. Given no a priori knowledge about the channel, the safest approach is to fix the middle tap as constant.

Note that the solution of the *Problem 2* is equivalent to the solution of *Problem 1* which is the minimization of a convex cost function of $\{c_n\}$, the combined equalizer and channel impulse response, under a linear constraint on $\mathbf{w}$.
Placing the FIR constraint on the equalizer coefficients will still preserve the convex nature of the problem. However, there will be a performance degradation due to this constraint.

3.3.3 l_p Norm As Convex Cost Function (Vembu Algorithm)

Since l_∞ norm is not differentiable and not suitable for gradient search. Instead of using this directly Vembu [14] proposed the use of l_p norm with a large p value. By making this assumption the convexity of the cost function is preserved while differentiability is satisfied. However using this cost function increases the complexity of the algorithm since power and root operations take place. Moreover the need for using a large enough p causes finite precision problems which is another drawback of this cost function. Finally, the solution of this approximation will not be a correct equalization point especially for small p values.

3.3.4 Linear Programming Based Methods

It is well-known that [16] minimizing the infinity norm of affine functions can be posed as linear programming problems(LP). Indeed this approach has been proposed for blind equalization of both symbol spaced and fractionally spaced channels. The real time implementation requirement of lp approaches places a limit on the data window used by the adaptive algorithms. In order to reflect this requirement, we can modify the previous convex optimization problem(*problem 2*) as

$$\begin{aligned} &\text{minimize } ||z \odot r||_\infty \qquad \textit{(Problem 3)} \\ &\text{s. t. } w_L = 1 \end{aligned}$$

where $\{r_n\}$ is a rectangular window function, with

$$r_n = \begin{cases} 1 & 0 \le n \le \Omega - 1, \\ 0 & \text{otherwise}, \end{cases} \tag{3.35}$$

and $z \odot r$ represents the element by element multiplication of two discrete time sequences (i.e., $(z \odot r)_k = z_k r_k$). Hence the final problem would be equivalent to the minimization of the l_∞ norm of the finite size vector

$$\underbrace{\begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_{\Omega-1} \end{bmatrix}}_{\mathbf{z}} = \underbrace{\begin{bmatrix} y_0 & y_{-1} & \cdots & y_{-N_w+1} \\ y_1 & y_0 & \cdots & y_{-N_w+2} \\ \vdots & \ddots & \ddots & \vdots \\ y_{\Omega-1} & y_{\Omega-2} & \cdots & y_{\Omega-N_w} \end{bmatrix}}_{\mathbf{Y}} \underbrace{\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{N_w-1} \end{bmatrix}}_{\mathbf{w}} \tag{3.36}$$

Since $\mathbf{z}$ is a linear function of the search vector $\mathbf{w}$ and the constraint $w_L = 1$ is a linear constraint, the corresponding l_∞ norm minimization problem can be casted to the well-known Linear Programming (LP) formulation:

$$\begin{aligned} &\text{minimize } t \qquad \textit{(Problem 4)} \\ &\text{s.t.} \begin{bmatrix} -\mathbf{1} & \mathbf{\Gamma} \\ -\mathbf{1} & -\mathbf{\Gamma} \end{bmatrix} \begin{bmatrix} t \\ \mathbf{w_s} \end{bmatrix} \le \begin{bmatrix} -\mathbf{q} \\ \mathbf{q} \end{bmatrix} \end{aligned}$$

where $\mathbf{\Gamma}$ is equivalent to $\mathbf{Y}$ matrix with $(L+1)^{th}$ column deleted, $\mathbf{q}$ is the $(L+1)^{th}$ column of $\mathbf{Y}$, $\mathbf{w}_s$ is the $\mathbf{w}$ with $(L+1)^{th}$ element deleted (since $w_L = 1$), and $\begin{bmatrix} t & \mathbf{w_s}^T \end{bmatrix}^T$ is the search vector.

For the fractionally spaced case the LP formulation is very similar to the symbol spaced case. The main difference is that we have multiple subchannels corresponding to the different phases of observations as in Figure (1.2) hence the formulation is based on using all of these outputs. For each subchannel i in Figure (1.2) we have the equation for the output window z_i as follows:

$$\underbrace{\begin{bmatrix} z_{i0} \\ z_{i1} \\ \vdots \\ z_{i\Omega-1} \end{bmatrix}}_{\mathbf{z}_i} = \underbrace{\begin{bmatrix} y_{i0} & y_{i-1} & \cdots & y_{i-N_w+1} \\ y_{i1} & y_{i0} & \cdots & y_{i-N_w+2} \\ \vdots & \ddots & \ddots & \vdots \\ y_{i\Omega-1} & y_{i\Omega-2} & \cdots & y_{i\Omega-N_w} \end{bmatrix}}_{\mathbf{Y}_i} \underbrace{\begin{bmatrix} w_{i0} \\ w_{i1} \\ \vdots \\ w_{iN_w-1} \end{bmatrix}}_{\mathbf{w}_i} \tag{3.37}$$

For each of the channels we can write a similar expression. As the output of the equalizer is the sum of the multiple branches in Figure(1.2), we can write:

$$\underbrace{\mathbf{z}_1 + \mathbf{z}_2 + \cdots + \mathbf{z}_p}_{\mathbf{z}} = \underbrace{\begin{bmatrix} \mathbf{Y}_1 & \mathbf{Y}_2 & \cdots & \mathbf{Y}_p \end{bmatrix}}_{\mathbf{Y}} \cdot \underbrace{\begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_p \end{bmatrix}}_{\mathbf{w}} \tag{3.38}$$

In (3.38) the fractionally spaced equivalent of $\mathbf{w}$, $\mathbf{Y}$ and $\mathbf{z}$ in (3.36) are shown. It is clear from this expression that the channel is equalized using a linear combination of the p subequalizers. The rest of the LP formulation for fractionally spaced equalizers is the same as the one given in (3.36).

The linear constraints however, are slightly different for this case. For this new formulation, the combined response of the channel can be written as follows:

$$\mathbf{Hw} = \mathbf{c} \tag{3.39}$$

where H is defined by:

$$\mathbf{H} = [\mathbf{H}_1\mathbf{H}_2\cdots\mathbf{H}_p]. \tag{3.40}$$

Above $\mathbf{H}_i$ corresponds to the convolution matrix of the i^{th} outputs channel response. For a detailed formulation please refer to [39]. From the properties of the convolution, $\mathbf{H}_i$ is a tall matrix. This means $\mathbf{H}_i^T$ has a nonempty null space making exact equalization not possible for all the channels. For the fractionally spaced case we have p outputs forming the matrix $\mathbf{H}$. As we can see this time we have a fat matrix and from the structure of the convolution matrix(p toeplitz submatrices) we can say that the matrix has full column rank. This allows us to equalize the channel exactly. However this time we have more problem having the zero output at points other then the all-zero setup since the matrix has a nonempty null-space. Putting one linear constraint such as making an element of the equalizer equal to a constant, does not prevent the zero solution for most of the time. Since the matrix fat it is very likely that the rest of the matrix(columns other then the constrained one) is full column rank hence zero output vector is still a solution. To overcome this problem a second constraint is defined. That is equating a portion of the equalizer vector $\mathbf{w}$ to zero. The number of elements that are made zero is equal to the difference between the number of columns and the number of rows. The columns corresponding to the nonzero elements of the matrix $\mathbf{H}$ can be assumed to be independent. Now putting the previous linear constraint on the unconstrained portion of the equalizer vector will clearly force the algorithm to avoid the zero output and the all-zero solution. The constrainted vector $\mathbf{w}$ is as follows:

$$\mathbf{w} = [w_1 \quad \ldots \quad w_{N_w+N_h+1} \quad 0 \quad \ldots \quad 0]^T , \tag{3.41}$$

where $w_k = c$ and $0 \leq k \leq N_w + N_h + 1$.

Although LP formulation provides exact solutions to finite window symbol spaced and fractionally spaced cases, obtaining the solution of an LP is a demanding task. In the next chapter we introduce our approach where we use subgradient optimization to develop low complexity algorithms with the same goal. The examples in chapter 6 show that nearly the optimal solution is achieved with much lower complexity then the LP based approaches.

Chapter 4

FAST AND LOW COMPLEXITY BLIND EQUALIZATION VIA SUBGRADIENT PROJECTIONS

4.1 SGBA

In order produce low complexity iterative algorithms for solving the *Problem 3*, we apply the subgradient projection approach which is outlined in chapter 2. We start by writing the subdifferential set for the blind equalization cost function

$$f(\mathbf{w}) = \|\mathbf{z}\|_\infty = \|\mathbf{\Gamma}\mathbf{w}_s + \mathbf{q}\|_\infty, \tag{4.1}$$

which is given by

$$\begin{aligned} \partial f(\mathbf{w}_s) \;\; = \;\; & \mathbf{Co}\Big\{\{\mathbf{\Gamma}_{k,:}^T | z_k = ||\mathbf{z}||_\infty\} \cup \\ & \{-\mathbf{\Gamma}_{k,:}^T | z_k = -||\mathbf{z}||_\infty\}\Big\}, \end{aligned} \tag{4.2}$$

where $\mathbf{Co}\{\}$ is the convex-hull operation and $\mathbf{\Gamma}_{k,:}^T$ is the k^{th} row of the matrix $\mathbf{\Gamma}$ which corresponds to the equalizer input vector producing output z_k. Here we obtained the subdifferential set by using the following properties of subgradients:

- Given a set of convex functions $h_i(\mathbf{x})$, $i = 1 \ldots m$, with the corresponding subdifferential sets $\partial h_i(\mathbf{x})$, if

  $$h(\mathbf{x}) = \max_{i=1\ldots m} h_i(\mathbf{x}), \tag{4.3}$$

  then

  $$\partial h(\mathbf{x}) = \mathbf{Co}\{\bigcup\{\partial h_i(\mathbf{x}) \;\; | \;\; h_i(\mathbf{x}) = h(\mathbf{x})\}\}, \tag{4.4}$$

  where **Co** represents the convex hull operation.

- Given $f(\mathbf{x}) = h(A\mathbf{x} + b)$, where h is convex, then $\partial f(\mathbf{x}) = \{A^H\mathbf{y} \;\; | \;\; \mathbf{y} \in \partial h(\mathbf{x})\}$.

Upon the inspection of the subdifferential set in Equation 4.2, it can be seen that the search direction for the subgradient projection algorithm is obtained from the equalizer

input vectors causing the maximum magnitude equalizer output within the given window. If J is the set of the time instants for which the maximum magnitude is achieved, i.e., $J = \{k \mid |z_k| = ||\mathbf{z}||_\infty\}$, then a possible search direction for the subgradient projection algorithm is given by

$$\mathbf{m} = -\sum_{k \in J} \xi_k \text{sign}(z_k) \mathbf{\Gamma}_{k,:}^T \tag{4.5}$$

where $\sum_{k \in J} \xi_k = 1$ and $\xi_k \geq 0$. For convenience, one may choose $\xi_l = 1$ for some $l \in J$ and $\xi_k = 0$ for $k \neq l$ in which case the search direction simplifies to

$$\mathbf{m} = -\text{sign}(z_l) \mathbf{\Gamma}_{l,:}^T. \tag{4.6}$$

As a result, we can write the update rule for the subgradient based blind equalization algorithm (SGBA) as

$$\mathbf{w}_s^{(i+1)} = \mathbf{w}_s^{(i)} - \mu^{(i)} \text{sign}(z_{l^{(i)}}^{(i)}) \mathbf{\Gamma}_{l^{(i)},:}^T, \tag{4.7}$$

where

- $l^{(i)} \in \{0, ..., \Omega - 1\}$ is the index where maximum magnitude output is achieved at the i^{th} iteration.
- $\mu^{(i)}$ is the step size at the i^{th} iteration.

4.1.1 Step Size Selection for SGBA

There are various possible choices for $\mu^{(i)}$:

- **Constant step size** ($\mu^{(i)} = \mu$): This is the simplest choice for the step size. The small values of μ will cause slow convergence and the high values will cause high misadjustment, where we call the misadjustment as the percent deviation from the optimal value. As shown in [22], for the constant step size rule

$$\lim_{i \to \infty} f_i \leq f^* + \frac{\mu C^2}{2} \tag{4.8}$$

where C is an upperbound for the norm of subgradients as described in the previous section. We can set C as

$$C = \max_i \|\mathbf{\Gamma}_{i,:}\|_2 \tag{4.9}$$

in our problem, as the subgradients are the rows of the $\mathbf{\Gamma}$ matrix. Therefore, for a given constant step size μ the distance to the optimal point is always bounded and there is no issue of unstable error growth, due to the fact that the norm of the subgradient is always bounded.

- **Normalized Constant Step Size** ($\mu^{(i)} = \frac{\mu}{\|\mathbf{g}^{(i)}\|_2}$): This is in analogy to Normalized-LMS (NLMS) where the step size has a dynamic adjustment based on the norm of the current search vector.

- **A Zero-Limit-Divergent-Sum(ZLDS) Step Size** ($\mu^{(i)} = \frac{\mu_0}{i+1}$): This step size rule satisfies the Polyak's Conditions in 2.10 and 2.11 and therefore the SGBA using this step size is guaranteed to converge (to the solution of *Problem 3*) as we mentioned in the previous section.

- **The Relaxation Rule with Variable Target Value:**

$$\mu^{(i)} = \alpha \frac{|z_{l^{(i)}}^{(i)}| - \hat{f}^{*(i)}}{\|\mathbf{\Gamma}_{l^{(i)},:}\|_2^2}, \tag{4.10}$$

as in the relaxation rule of Equation 2.14 with $\alpha \in (0, 2)$. If the target value f^* were known a priori then this scheme would converge with a monotonic decrease in distance to the solution of *Problem 3*. However, as f^* is not known a priori, we need to use a variable target value scheme. There are different approaches in updating $\hat{f}^{*(i)}$:

  - Use of variable target schemes outlined in the previous section [30, 22, 31]. Among these algorithms, we used the Goffin&Kiwiel algorithm[30] in our simulations in Section 6. This algorithm has a relatively simple rule for updating the target value $\hat{f}^{*(i)}$ and it is guaranteed to converge to f^*.
  - As an alternative variable target scheme, for constant modulus signaling we propose the use of

$$\hat{f}^{*(i)} = \frac{1}{(\Omega)^{1/p}} \|\mathbf{\Gamma}\mathbf{w}_s^{(i)} + \mathbf{q}\|_p, \qquad p = 1, 2 \tag{4.11}$$

which is the average of the magnitude of the output for $p = 1$ or the RMS of the output for $p = 2$ within the selected window. The resulting μ_i will always be

nonnegative since

$$\frac{1}{(\Omega)^{1/p}}\|\mathbf{\Gamma w}_s^{(i)} + \mathbf{q}\|_p \leq \|\mathbf{\Gamma w}_s^{(i)} + \mathbf{q}\|_\infty \tag{4.12}$$

with equality if the magnitude of the equalizer output is constant which refers to the perfect equalization condition. For non-constant modulus constellations proper adjustments can be made to $\hat{f}^{*(i)}$ by scaling with a constellation dependent parameter.

With the choice of the variable target value the algorithm becomes independent of the information about the cost function which is not available for most of the time.

4.2 Fixed Window SGBA

If we keep the data matrix Γ to be the same at each iteration of SGBA algorithm given by (4.7), it would be equivalent to the use of some data window. Therefore, we refer to this case as the Fixed Window SGBA.

4.3 Moving window SGBA

The fixed window SGBA algorithms performs very well in environments with a low noise level. As seen in the results, its speed is independent from the noise level however the convergence precision is suffered from the noise. Also when the channel is changing we want to track the channel. With a fixed window this is not possible. When the change in the channel makes equalization impossible, the only thing we can do is to form a new window from newer samples and to restart the algorithm. Finally there are some cases where we can prefer lower complexity for higher convergence speeds. These problems can be summed up as the following two statements:

- The complexity of the algorithm is linearly dependent to the size of the window hence if the size of the window is reduced then the complexity of the algorithm will also reduce linearly. In situations where the output of the equalizer is calculated in the hardware this complexity is not a concern. However if the output of the equalizer is

calculated via software then this will have a significant effect on the complexity of the algorithm. (*Problem 4*)

- Fixed Window SGBA is not robust against noise since the window is static. Although the eye opening performance is independent of noise, later in the equalization process we end up with oscillation around the optimal point(The range of the oscillation is linearly dependent to the variance of the noise). (*Problem 5*)

- Since the window is static when a change in the channel occurs the equalizer can not track the channel. (*Problem 6*)

To overcome these problems we introduce the SGBA with a moving window. The main idea is to use a dynamic window with a much smaller size instead of a large fixed window for estimating the infinity norm of the output. The dynamic window will make it possible to reduce the effect of the noise since the noise is independent in time and the average of the noise eventually goes to zero.
The dynamic window not only consists of new data vectors but also have some of the old data vectors. In other words this window has two main sub matrices having the old and the new vectors. If we define Γ^k as the dynamic window then the two main components can be written as follows:

$$\Gamma^k = \begin{bmatrix} \Gamma_1^k \\ \Gamma_2^k \end{bmatrix}. \tag{4.13}$$

In the equation above:

- $\Gamma_1^k \in C^{R_1 \times (N_w - 1)}$ consists of the subgradient vectors from previous iterations.

- $\Gamma_1^k \in C^{R_2 \times (N_w - 1)}$ consists of the subgradient vectors from the current iteration.

(Remember that every input vector to the window matrix is also a subgradient vector for the cost function).
As seen above Γ_1^k matrix serves the window as a memory element since it covers previous subgradients. This memory is crucial since the l_∞ norm is dependent on the maximum

element in the whole time interval.

The update rule for the moving window SGBA is very similar to the Fixed window SGBA and is given as follows:

$$\mathbf{w}^{(i+1)} = \mathbf{w}^{(i)} - \mu^{(i)} sign(z_{l^{(i)}}^{(i)}) \Gamma_{l^{(i)}}^{(i)^T}. \tag{4.14}$$

The only difference of this update rule is that the subgradient vector is chosen from the moving window instead of a static one. The crucial point in the algorithm is the update rule for the moving window. The update procedure is given as follows:

- $\Gamma_1^{(k+1)}$ is formed by N vectors in $\Gamma^{(k)}$ giving the closest values to the value of the cost function.
- $\Gamma_2^{(k)}$ matrix is formed by the newly acquired subgradient vectors.

When we have Γ_2^k having length L, we can write this matrix in terms of y_i (input of the equalizer) as follows:

$$\Gamma_2^k = \begin{bmatrix} y_{L\times k} & y_{L\times k+1} & \cdots & y_{L\times k+N_w-2} \\ y_{L\times k+1} & y_{L\times k+2} & \cdots & y_{L\times k+N_w-1} \\ \vdots & \vdots & \cdots & \vdots \\ y_{L\times k+P-1} & y_{L\times k+P} & \cdots & y_{L\times k+N_w+P-1} \end{bmatrix}. \tag{4.15}$$

The formulation given above is simulated under noisy conditions and the simulation results can be found in the chapter 6. The suggested method reduces the complexity while keeping the speed of convergence in reasonable bounds. Also it will be seen that the algorithm has gained robustness against noise. The discussion of the results will be given after the simulation results.

This concludes the "Moving Window SGBA " algorithm. Now we can move on to "Weighted SGBA" which offers a great improvement in speed of convergence for SGBA algorithms.

4.4 Weighted SGBA

We introduce the following modification to the update rule in Equation 4.7

$$\mathbf{w}_s^{(i+1)} = \mathbf{w}_s^{(i)} - \mu^{(i)} \mathbf{\Pi}^{-1} \mathbf{g}^{(i)}, \tag{4.16}$$

where $\mathbf{g}^{(i)} = \text{sign}(z_{l(i)}^{(i)}) \mathbf{\Gamma}_{l(i),:}^T$ is the subgradient. The only change in Equation 4.16 compared to the original SGBA is the introduction of the weighting matrix $\mathbf{\Pi}^{-1}$, which is constrained to be positive definite. If we choose the step size as

$$\mu^{(i)} = \gamma_i \frac{|z_{l(i)}^{(i)}| - f^*}{\|\mathbf{\Pi}^{-1}\mathbf{g}^{(i)}\|_{\mathbf{\Pi}}^2}, \tag{4.17}$$

where $0 < \gamma_i < 2$, then we can show that, given $f^{(i)} \neq f^*$,

$$\|\mathbf{w}_s^{(i+1)} - \mathbf{w}_s^*\|_{\mathbf{\Pi}} < \|\mathbf{w}_s^{(i)} - \mathbf{w}_s^*\|_{\mathbf{\Pi}}, \tag{4.18}$$

i.e. the weighted distance to the optimal point decreases monotonically. In order to prove this statement, we write

$$\begin{aligned}
\|\mathbf{w}_s^{(i+1)} - \mathbf{w}_s^*\|_{\mathbf{\Pi}} &= \|\mathbf{w}_s^{(i)} - \mathbf{w}^* - \mu^{(i)} \mathbf{\Pi}^{-1} \mathbf{g}^{(i)}\|_{\mathbf{\Pi}} && (4.19)\\
&= \|\mathbf{w}_s^{(i)} - \mathbf{w}^*\|_{\mathbf{\Pi}}^2 - 2\mu^{(i)} \mathbf{g}^{(i)^T} (\mathbf{w}_s^{(i)} - \mathbf{w}_s^*) + {\mu^{(i)}}^2 \|\mathbf{\Pi}^{-1}\mathbf{g}^{(i)}\|_{\mathbf{\Pi}}^2 && (4.20)\\
&\leq \|\mathbf{w}_s^{(i)} - \mathbf{w}_s^*\|_{\mathbf{\Pi}}^2 - 2\mu^{(i)} (|z_{l(i)}^{(i)}| - f^*) + {\mu^{(i)}}^2 \|\mathbf{\Pi}^{-1}\mathbf{g}^{(i)}\|_{\mathbf{\Pi}}^2 && (4.21)\\
&= \|\mathbf{w}_s^{(i)} - \mathbf{w}_s^*\|_{\mathbf{\Pi}}^2 - \gamma_i(2-\gamma_i) \frac{(|z_{l(i)}^{(i)}| - f^*)^2}{\|\mathbf{\Pi}^{-1}\mathbf{g}^{(i)}\|_{\mathbf{\Pi}}^2} && (4.22)
\end{aligned}$$

where the inequality in (4.21) is obtained by using the fact that $\mathbf{g}^{(i)}$ is a subgradient and therefore we can use the inequality in Equation 2.3. For $0 < \gamma_i < 2$ the second term in (4.22) is strictly positive and therefore we obtain the inequality in (4.18). As a result, since the weighted distance to the optimal point decreases monotonically and it is lower bounded by 0, the convergence to the optimal point is achieved in the limit.

Note that this proof of convergence with monotonic decrease in weighted distance to the optimal point is valid for any positive definite $\mathbf{\Pi}$. If we choose $\mathbf{\Pi} = \mathbf{I}$ then the weighted SGBA would be equivalent to the ordinary SGBA algorithm with relaxation step size rule and therefore we have a proof of convergence covered for this case.

Another choice of interest is the selection of $\mathbf{\Pi}$ as the covariance of the observations vector,

$$\mathbf{y}_i = \begin{bmatrix} y_i & y_{i-1} & \cdots & y_{i-L+1} & y_{i-L-1} & \cdots & y_{i-N_w+1} \end{bmatrix}^T, \tag{4.23}$$

i.e.,

$$\mathbf{\Pi} = R_{\mathbf{y}} \tag{4.24}$$

$$= E(\mathbf{y_i}\mathbf{y_i}^T) \tag{4.25}$$

which would yield

$$\|\mathbf{w}_s^{(i)} - \mathbf{w}_s^*\|_{\mathbf{\Pi}} = (\mathbf{w}_s^{(i)} - \mathbf{w}_s^*)^T R_{\mathbf{y}} (\mathbf{w}_s^{(i)} - \mathbf{w}_s^*) \tag{4.26}$$

$$= E(|\mathbf{w}_s^{(i)^T}\mathbf{y}_i - \mathbf{w}_s^{*T}\mathbf{y}_i|^2) \tag{4.27}$$

which is the mean square of the error between the outputs of a given $\mathbf{w}^{(i)}$ and the optimal equalizer $\mathbf{w}^*$. Therefore, if the covariance matrix of the observations vector is estimated and its inverse is used as the weighting matrix, then the corresponding weighted SGBA will provide a monotonic decrease in the output mean square error relative to the optimal equalizer $\mathbf{w}^*$. Furthermore, the update rule in Equation 4.16 will mimic the Newton algorithm.

4.5 Fractionally Spaced SGBA

The SGBA algorithm can be easily adapted to the fractionally spaced equalization setting in Figure (1.2). Although the treatment is very similar to the standard SGBA case there are some important differences.

First and the most important difference is this time we assume to have a full rank, fat matrix allowing us to have perfect equalization. Under some generic assumptions about the channel perfect equalization will occur at the points (excluding 0) where the cost function is equal to the average value of the absolute values of the equalizer outputs($J(\mathbf{w}) = \|\mathbf{z}\|_\infty$). This is the case when all the outputs have the same magnitudes and this is only possible when the combined response is equal to a scaled and shifted version of delta function.

Another difference is that we will avoid using any constraints on the equalizer vector. Without any linear constraints the cost function becomes

$$f(\mathbf{w}) = \|\mathbf{z}\|_\infty = \|\mathbf{\Gamma w}\|_\infty. \tag{4.28}$$

The target value estimate function f_{avg}

$$f_{avg}(\mathbf{w}) = \frac{1}{(\Omega)}\|abs(\mathbf{\Gamma w})\|_1, \tag{4.29}$$

For this case the global minimum of the cost function will surely be equal to 0 and this value will be reached at the origin. In Figure (4.1) the epigraph of the cost and the averaging function on a line **a** passing from the origin is shown. At any point on the **a** axis the subgradient vector is the same. This subgradient vector $\mathbf{g}^i$ is the vector giving the max $|< \mathbf{g}_j \cdot \hat{\mathbf{a}} >|$ and linearity guarantees us that along axis **a** this product stays maximum among all of the subgradients. Since we are considering the absolute value of the outputs and no linear constraints, it is clear that the cost function f_{cost} is symmetric with respect to the origin. Since f_{avg} is formed by a weighted sum of these subgradient vectors (and there are no constraints)it is also symmetric.

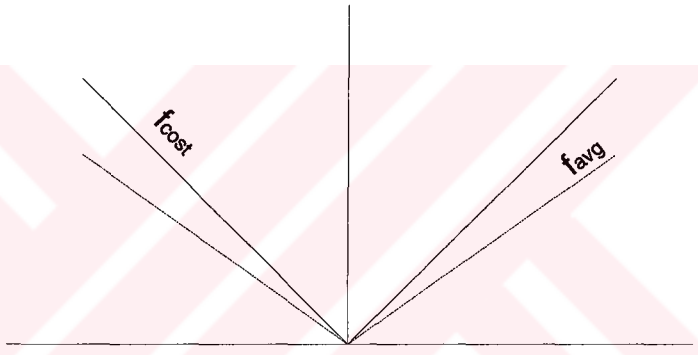

Figure 4.1: Cost function on an arbitrary line crossing from the origin.

About the average and the cost functions, the statements below can be written:

$$f_{avg} \neq f_{cost}. \tag{4.30}$$

If $f_{avg}(x) = f_{cost}(x)$ where $x \neq 0$, then at any point y belonging to the axis passing from the origin and x,

$$f_{avg}(y) = f_{cost}(y) \tag{4.31}$$

since both functions epigraph for an axis a is nothing more then a simple line (4.1) and both functions have the same value at the origin. With this new x point we have two lines intersecting at two points and this is possible if and only if two lines are the same.
We have stated that the points where (other then the origin) the cost and the averaging functions are equal to each other, are the equalizing points(points coordinates are the equalizer

vector's coefficients). Now we can easily say that in the space containing the subgradient vectors there are lines passing from the origin which are formed by the equalizing points. As the equalizer vector gets close to the neighborhood of one of these lines, the cost function gets closer to the average function. The step size approaches to 0 as the equalizer vector gets closer to one of these line(s). Then it is clear that when we get closer to this line the algorithm's step size will get smaller and eventually when we reach the line it reaches zero causing the algorithm to stop. When the algorithm stops at a point other then the origin, although this is not the global minimum for this function this is a point which equalizes the channel.

Without the linear constraint we can see that there are two possible outcomes of this algorithm. One of them is to reach an equalizing point and the other is to reach the origin which is the all-zero setup.

Although with finite precision effects it is nearly impossible to reach the origin by using SGBA, theoretically it is possible. Suppose that the algorithm reaches to a point x where $x = k \times \mathbf{g}^i$ and at point x the subgradient is equal to $\mathbf{g}^i$. Also assume that the cost function is not equal to the averaging function at point x. Since the direction of $\mathbf{x}$ and $\mathbf{g}^i$ are the same, the next point will also be located on the line between the origin and point x. From the fact that the subgradient of the cost function is the same on every point along a line crossing the origin , clearly $\mathbf{g}^i$ will be the subgradient choice for the rest of the iterations and since this is not an equalizing line then the algorithm will converge to the origin. In figure (4.2) an example for this case can be seen.

This makes the choice of the initial point very crucial(In the simulations it can be seen that with a wise selection of the initial point, reaching the origin is very unlikely).

The convergence to zero can be avoided by renormalization of $\mathbf{w}$ every time $\|\mathbf{w}\|_2$ drops below a certain level. By this scheme only possible stationary points of the algorithm would be the equalizing points.

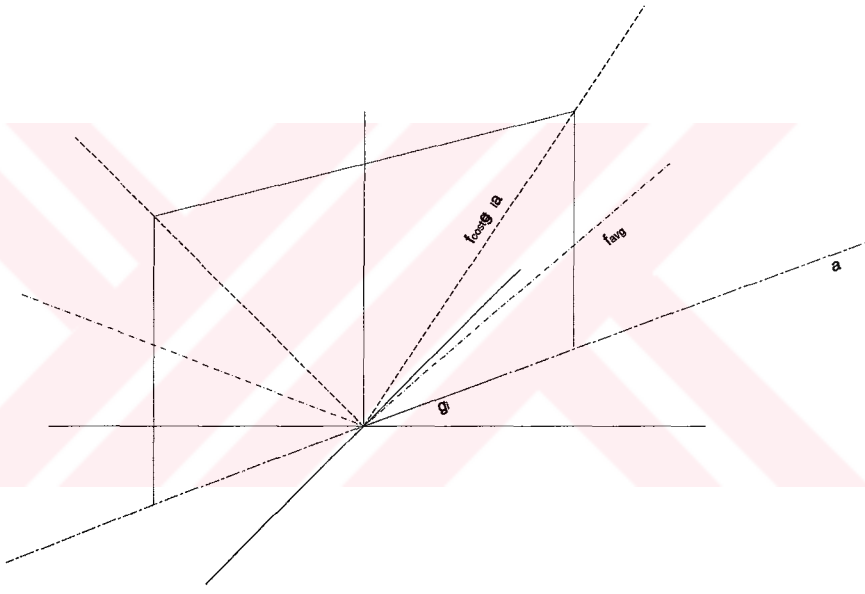

Figure 4.2: In this case SGBA algorithm converges to the origin directly.

Chapter 5

COMPLEXITY ANALYSIS

The basic advantage of SGBA is its simple update rule (4.7), which makes it suitable for hardware-only implementation. Here the update vector is calculated from the inputs and the outputs of the equalizer with a simple mathematical operation and therefore the algorithm is less prone to errors due to fixed point implementations.

In SGBA, for each iteration, we need to

- compute Ω output samples, which requires ΩN_w multiplications and $\Omega(N_w - 1)$ additions,
- determine the peak location, which requires $\Omega - 1$ subtractions,
- compute the step size whose complexity depends on the step size rule:
  - the constant step size: no computation,
  - ZLDS: one division.
  - Variable target scheme with $\hat{f^*}^{(i)}$ in Equation 4.11: for the average magnitude case (p=1), N_w multiplications, $\Omega + N_w - 1$ additions and and 2 divisions.
- compute the update, which requires N_w multiplications and N_w additions.

Overall we need about $(\Omega + 1)N_w$ multiplications and additions (and additional computational load depending on the step size selections scheme) per iteration. Therefore, the overall complexity is $(\Omega + 1)N_w\eta_{SGBA}$ operations where η_{SGBA} is the number of iterations. The linear programming solution of *Problem 3* requires $O(\Omega^3 L)$ operations, where L is a precision based factor[40]. Therefore, considering that the number of iterations is typically much less than the window length, as justified by the examples in the next section, there is a big computational saving obtained by the SGBA algorithm compared to the linear programming solution.

The major computational burden of SGBA is due to the output computation which requires $\Omega N_w \eta_{SGBA}$ operations whereas the update part requires (assuming simple step size selection rule) only $N_w \eta_{SGBA}$ ($N_w^2 \eta_{SGBA}$ in weighted SGBA) operations. In many physical layer communication system implementations, the equalizer is designed to be the part of the hardware, and therefore both input and output samples are available to the adaptive algorithm during the data collection stage with no additional cost. As a result, in such systems, unless there is a constraint on the reuse of the same input data window, we can drop the load of the output computations from the overall computation budget. The peak detection (and the magnitude averaging required for Equation 4.11) can be also done during the data collection stage with simple additions. At the end of the data collection, the update can be performed which requires only N_w (N_w^2 for weighted SGBA) operations per iteration.

If we compare the complexity SGBA with some other iterative methods's complexities:

- CMA algorithm[4, 5] requires $N_w \eta_{CMA}$ operations. Here $\eta_{CMA} >> \eta_{SGBA}$ in general.
- the super-exponential algorithm by Shalvi-Weinstein[36] requires $\Omega N_w \eta_{SW}$ operations for output computation, $6\Omega N_w \eta_{SW}$ operations for the computation of the empirical cumulant expression and $N_w^2 \eta_{SW}$ operations for the multiplication with the inverse covariance matrix. As illustrated in Section 6 η_{SW} is typically less than η_{SGBA}, however the overall computational complexity of both algorithms are similar.
- Vembu algorithm [14] (based on l_p approximation of l_∞) requires $N_w \eta_{Vembu} p$ and typically $p\eta_{Vembu} >> \eta_{SGBA}$.

When the output computation can be dropped from the computational budget by the hardware implementation of the equalizer, SGBA has significantly lower complexity than all of the above algorithms.

5.1 Moving Window SGBA

The moving windowed version of SGBA has a very similar expression for complexity as the fixed window SGBA. The difference is the additional complexity caused by the update of

the row vectors in the moving window. For each update, these vectors are sorted by the corresponding output value(The vectors giving the closest value to the cost function are selected). The complexity of the sort operation is $O(N_\Gamma \lg(N_\Gamma))$. Since N_Γ is very small for the moving window SGBA this addition is nearly negligible (as we will see in the simulations the dynamic window size is nearly 50 times smaller then the static ones size and this makes the number of iterations the dominating term for the complexity). As we will see the rate of increase in the number of iterations is less then the rate of decrease in the window size causing the moving window SGBA to have significantly lower complexity then fixed window SGBA.

5.2 Weighted SGBA

Weighted SGBA needs the calculation of the weighting matrix $\prod$ and $\prod^{-1}$. Since this calculation needs to be done only once it will not cause a burden on the complexity. Also for a fixed window, $\prod^{-1}\mathbf{g}^i$ and its norm can be precalculated only once (since $\mathbf{g}^i$ vectors are also predefined). We can say that for a static window the complexity of the weighted SGBA is nearly the same as the complexity of the standard SGBA.

5.3 Fractionally Spaced SGBA

For this case only the definition of the matrices (block convolution matrices rather then standard convolution matrices) are different but the algorithms update rule stays the same. As we can see the formula to calculate the complexity of the algorithm is the same as the fixed window SGBA. However the size of the matrix Γ is p times wider as we can see in equation (3.38). Then under the same conditions we can see that the complexity of the fractionally spaced equalizer will be p times higher then the standard SGBA per iteration.

Chapter 6

PERFORMANCE ANALYSIS OF SGBA

In this chapter, we provide examples illustrating the performance of the SGBA algorithm for some sample channels.

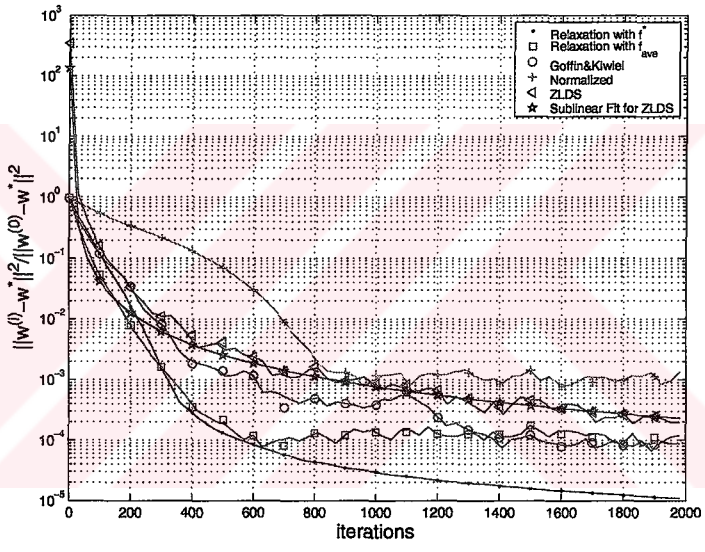

Figure 6.1: Distance to the optimal point for different step size selections.

6.1 Comparison Of Step Size Rules

In the first example, we compare the convergence performance of different step size selection rules. We assume

- $h = \{-1.0493 + 0.2305i, 1.4129 - 1.4497i, -0.2540 + 0.2021i, 0.5302 - 0.7732i\}$ as the impulse response of the channel (taken from the reference [41]),
- 4 QAM Input Constellation,

- $N_\omega = 21$, $L = 8$, $\Omega = 400$.

The step size selection schemes we compared are:

1. *Relaxation*: $\mu^{(i)} = 1.7\frac{z_{l(i)}^{(i)} - f^*}{\|\mathbf{\Gamma}_{l(i),:}\|_2^2}$ where f^* is assumed to be known. Here we pre-calculated f^* using the Linear Programming method. This is not reasonable in real-life applications, however, we include this case to provide a basis of comparison for the other step size selection schemes.

2. *Relaxation with Average*:$\mu^{(i)} = 1.3\frac{|Rez_{l(i)}^{(i)}| - \hat{f}^{*(i)}}{\|\mathbf{\Gamma}_{l(i),:}\|_2^2}$ where $\hat{f}^{*(i)}$ is selected as the average magnitude in Equation 4.11.

3. *Normalized Constant*: $\mu^{(i)} = \frac{0.013}{\|\mathbf{\Gamma}_{l(i),:}\|_2}$.

4. *ZLDS*: $\mu^{(i)} = \frac{10}{(i+1)\|\mathbf{\Gamma}_{l(i),:}\|_2}$.

5. *Goffin & Kiwiel Algorithm*: We updated the target level based on the rules presented in Section 2.3.3, following the steps of the Goffin & Kiwiel algorithm in [30]. Here we choose the threshold for the total path length traversed for the given target level as $0.9(R$ parameter in [30] and the initial distance between the minimum and the target level as 4 (δ_1 parameter in [30]).

The plots for the distance to the optimal point and the open eye measures for the different step size selection rules are shown in Figures 6.1 and 6.2 respectively. Among different choices, the best performance is achieved with the relaxation with perfect knowledge of f^*, for which the eye becomes open after 40-60 iterations. When f^* is replaced with the average magnitude approximation, a similar eye-opening performance is achieved. However, the performance is worse after the eye-opening stage. The degradation is due to fact that only a finite time window is considered and a sample spaced FIR equalizer is employed such that the perfect equalization condition is never achieved. If the SBGA algorithm is considered as the initial eye-opener algorithm, which opens the initially closed eye and then leaves the stage to the decision directed blind algorithms later, then the relaxation with average magnitude step size rule can replace the relaxation with f^* as their eye opening performances are almost identical.

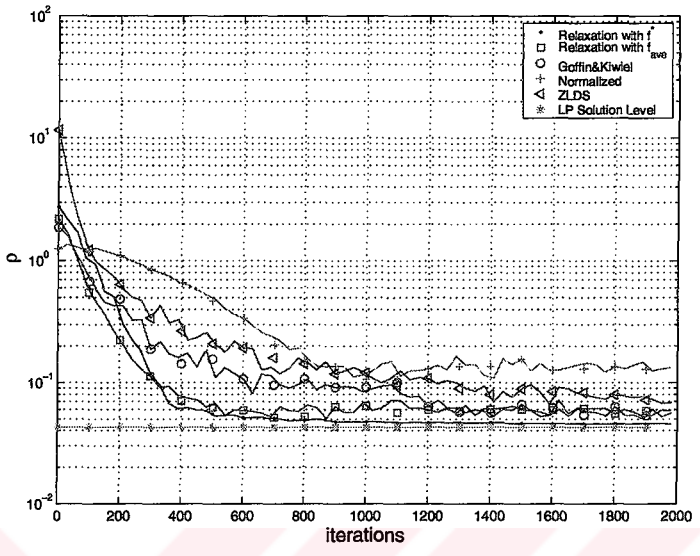

Figure 6.2: Open eye measure for different step size selections..

The ZLDS rule, although slower, provides a reasonably fast convergence. The expected sublinear convergence behavior of ZLDS (from Section 2.3.1) is also confirmed by Figure 6.1. The normalized constant step size rules leads to a slow convergence as expected. The Goffin&Kiwiel algorithm, after careful tune of its parameters, provides fast convergence. However, the performance of this algorithm tends to be very sensitive to the selection of its algorithm parameters. For example, if the threshold parameter for the expression in Equation 2.29 and the initial target value are not properly chosen, then the algorithm can become very slow. Similarly, we observed that the behavior of Nedic & Bertsekas algorithm is highly dependent on the β parameter in Equation 2.31 and slight deviations in β cause drastic differences in convergence behaviors.

6.2 Comparison of Blind Algorithms: Complex Channel

As another example, we consider the same complex channel as the previous example and we compare the convergence of SGBA with the following algorithms:

- CMA [5] with constant step size, where a high value of step size is chosen to increase

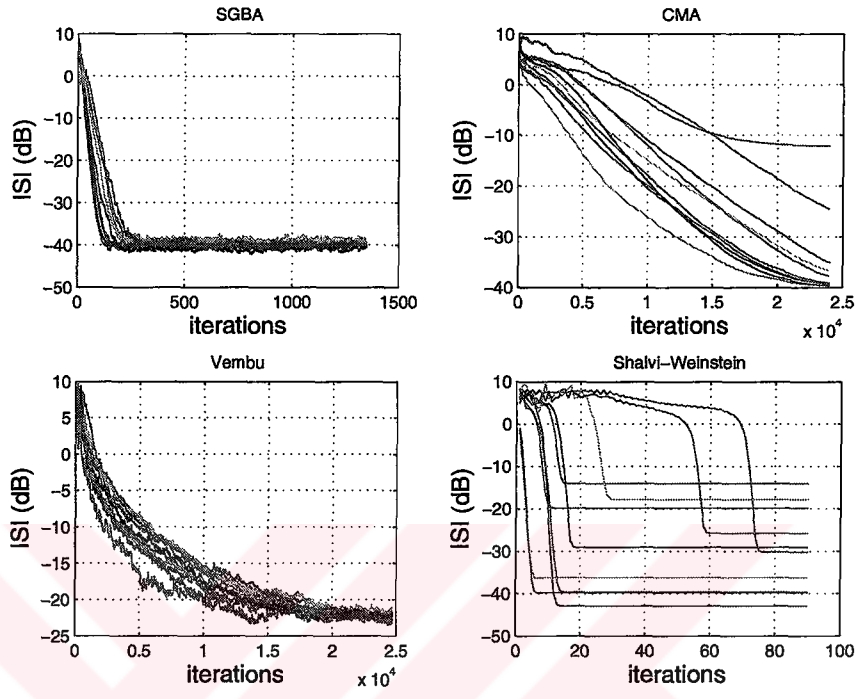

Figure 6.3: ISI as a function of iterations for the 4-tap complex channel.

the speed of convergence,

- Vembu Algorithm [14]: this is the l_p based approximation of the l_∞ cost function. We used a dynamic step size adjustment to increase the speed of convergence for the gradient search algorithm. We selected p parameter according to the following rule

$$p = \begin{cases} 8 & l < 5000 \\ 12 & 5000 \leq l < 9000 \\ 50 & 9000 \leq l \end{cases}, \tag{6.1}$$

  where l is the iteration index.

- Shalvi-Weinstein (super-exponential) Algorithm [36]. The window length is chosen as 400 (same as SGBA algorithm).

The Figure 6.3 summarizes the results of the simulations with all four algorithms, where each curve corresponds to a different initialization point. Initial points are chosen randomly from the surface of the sphere around the optimal point so that the initial distance to the optimal point is same at each run. In SGBA simulations, we used the SGBA with the relaxation rule with the average magnitude as the variable target value. The following are the observations based on this figure:

- convergence to a global optimal point is obtained by SGBA and Vembu algorithms whereas for Shalvi-Weinstein and CMA algorithms $\mathbf{w}$ can converge to different stationary points depending on the initialization[42]. The ISI level obtained by the Linear Programming based solution is $-39.6dB$, which verifies that the point converged by the SGBA algorithm is close to the solution of *Problem 3.*

- When entered into a steady convergence path Shalvi-Weinstein algorithm provide fast convergence. However, depending on the initial point, the performance may drag at high ISI levels for a while (which is probably caused by the attraction of saddle points).

- SGBA converges at reasonably low number of iterations and is less sensitive to the choice of initial point due to the convex nature of its cost function.

- Both CMA and Vembu algorithms are slow and require quite a large number of iterations.

6.3 Comparison of Blind Algorithms : DSL Channel

As the next example, we consider a DSL channel, the combination of ITU G.SHDSL Central Office transmitter mask and a transmission channel defined in standard test cases (CSA4 Channel with 2 bridge taps)[43]. The corresponding 128 tap impulse response and its frequency response are shown in Figure 6.4.

In this simulation, we assumed $M = 2$, $\Omega = 800$, $N_w = 30$, $L = 12$ and used the weighted SGBA (with average magnitude estimate of f^*). Figure 6.5 summarizes the results:

- The weighted SGBA algorithm converged in 50-to-100 iterations.

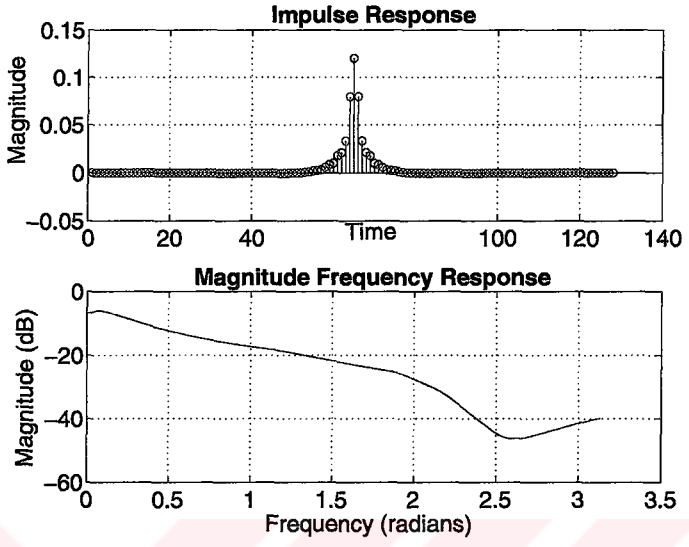

Figure 6.4: Impulse Response and the Frequency Response for a Sample DSL Channel.

- The super exponential algorithm convergence curves have initial high ISI-drag sections followed by sharp falls.

- The CMA and Vembu Algorithms again require large numbers of iterations. Furthermore, the stationary point of the CMA algorithm with -10dB ISI level corresponds to an undesired stationary point with two significant ISI taps (i.e., an undesired minima which is expected in a sample spaced FIR equalization setting.).

- The computations required for each algorithm:

  - SGBA: Assuming 75 iterations on average, the required computation is about 2 Million operations (1.8 Million for the output computations and 0.2 Million for the updates). Note that if the output computation's complexity can be dropped (with a hardware equalizer implementation) then the computational requirement reduces to 0.2 Million.

  - Shalvi-Weinstein: Assuming 20 iterations on average, the required computation is about 3.3 Million operations (0.5 Million for the output computations and 2.8

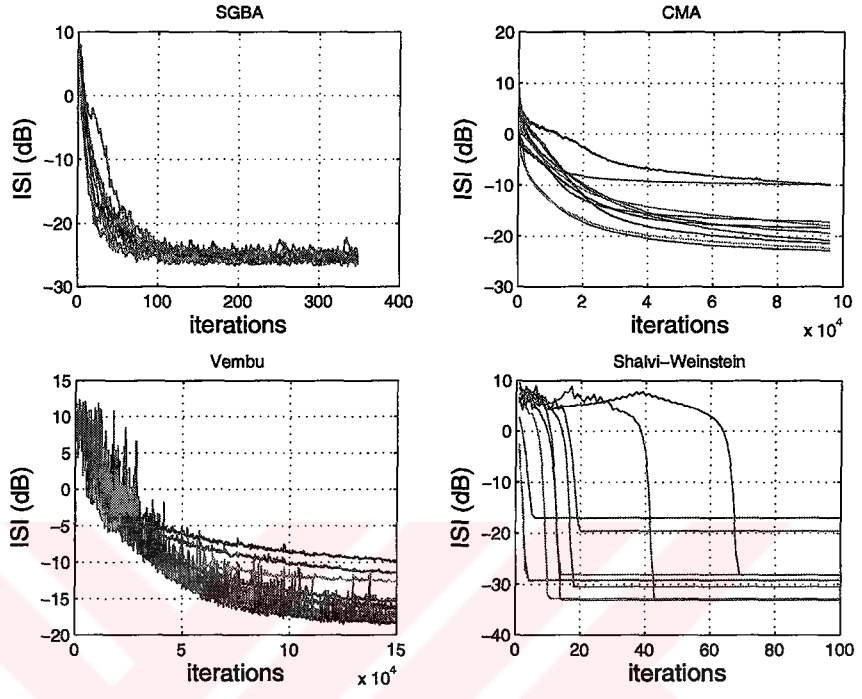

Figure 6.5: ISI as a function of iterations for the sample DSL channel.

Million for the updates)

- CMA: Assuming 10^5 iterations, the required computation is 6 Million operations.
- Vembu: The required computation is more than 20 Million operations.
- LP solution of *Problem 3*: In the order of 512 Million operations.

6.4 Moving Window SGBA

In this section we will look at some examples to illustrate the performance of the Moving Window SGBA. For the simulations in this section:

- $h = \{-1.0493 + 0.2305i, 1.4129 - 1.4497i, -0.2540 + 0.2021i, 0.5302 - 0.7732i\}$ as the impulse response of the channel (the same as the first example),
- In the moving window Γ^1 has size 7 and Γ^2 has size 1,

- the size of the equalizer is 17,
- this simulations are made for 5 different starting points which are chosen from a sphere that has a diameter equal to the norm of the optimum equalizer vector.
- the SNR value is $20dB$.

The ISI measure is used for the examples in this section(given in the introduction).

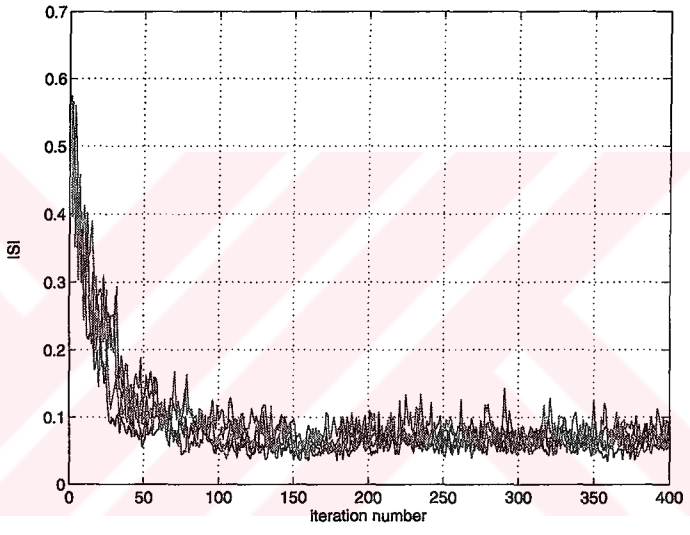

Figure 6.6: Fixed Window SGBA ISI as a function of iterations for the sample channel.

ISI is a reliable measure for eye opening performance. In the examples above, the channel eye begins to open near 1.5. In Mowing Window SGBA our aim is to open the channel eye. When the channel eye is open we can switch to decision directed equalization.
In figure 6.6 the ISI measure curve for a Fixed Window SGBA using window length 600 is given. It is seen that the algorithm is still independent from choice of initial point. The algorithm reaches 1.5 ISI value in nearly 50 iterations which is a very small number. However for 50 iterations we have $50 \times 600 = 30000$ number of output calculations. When the equalizer is implemented in hardware we don't have to worry about this computational burden but when this is not the case we simply face with a considerably high computational

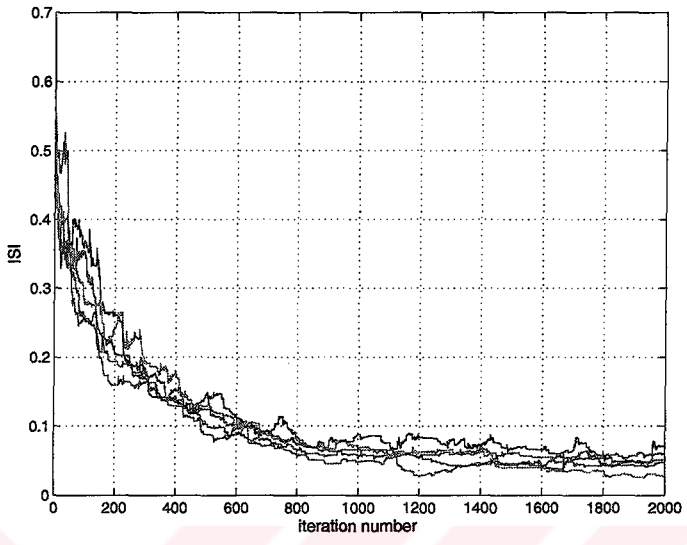

Figure 6.7: Moving Window SGBA ISI as a function of iterations for the sample channel.

complexity. Another important issue we see in 6.6 is the oscillations caused by the noise after the eye is opened.

In figure 6.7 the ISI curve for the Moving Window SGBA is given. It is seen that the dependence of the performance on the initial point selection is very low as the Fixed Window SGBA. The number of iterations for reaching the ISI value 1.5 is higher when compared with the Fixed Window SGBA. For reaching the target ISI value 1.5 nearly 350 iterations is needed. In this example our moving window size was 8 hence the average number of output calculations is $350 \times 8 = 2800$. When compared with the Fixed Window SGBA it is seen that we have a considerably lower complexity for this case.

6.5 Fractionally Spaced SGBA

In this section we will look at the simulation results for the Fractionally Spaced SGBA algorithm. For the simulations in this section:

- A fractionally spaced channel having single input and 5 outputs is used. The inpulse response of the channel is as follows:

$$h = \begin{bmatrix} -0.049 + 0.359i & 0.482 - 0.569i & -0.556 + 0.587i & 1 & -0.171 + 0.061i \\ 0.443 - 0.0364i & 1 & 0.921 - 0.194i & 0.189 - 0.208i & -0.087 - 0.054i \\ -0.221 - 0.322i & -0.199 + 0.918i & 1 & 0.284 - 0.524i & 0.139 - 0.19i \\ 0.417 + 0.030i & 1 & 0.873 + 0.145i & 0.285 + 0.309i & -0.049 + 0.161i \end{bmatrix}$$

- the size of the equalizer is 25,
- In the first simulation delta functions are used as the starting points of the algorithm. In the second simulation the points satisfying the scheme in (4.2) are used as the sataring points for the algorithm.

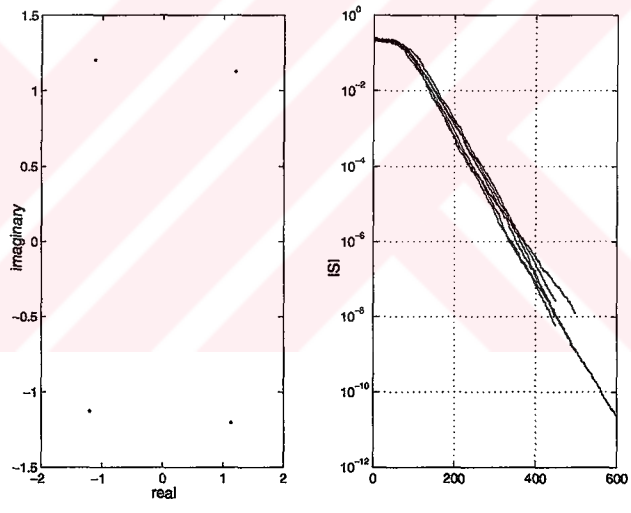

Figure 6.8: Fractionally spaced SGBA, delta function as the starting point.

In the first simulation (6.8) we can see that the algorithm gives similar results for different runs. Since the window size is large, it is obvious that the cost function and the averaging function will have similar characteristics for each run(for the same or scaled version of the starting point). The 1.5 ISI value is reached nearly 60 iterations which is pretty fast. Also you can see that perfect equalization is reached. First one (speed) is a property of SGBA and perfect equalization is the result of the setup being fractionally spaced.

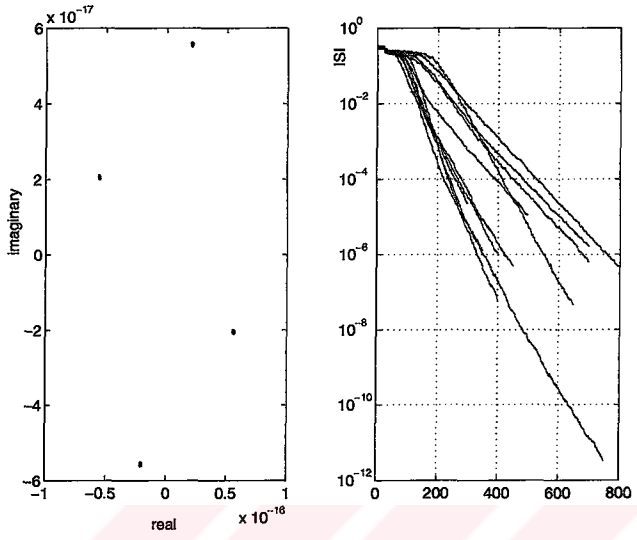

Figure 6.9: Fractionally spaced SGBA, starting point chosen for converging to zero.

In the second simulation (6.9) we can see that perfect equalization is reached for all of the simulations. Theoretically in these simulations the algorithm must converge to the zero point without hitting any of the equalizing points. However as we have stated before when the algorithm gets closer to 0 finite precision effects causes the algorithm to choose a wrong vector as subgradient. This change in direction results in the convergence to an equalizing point. You can see that the equalized constellation has a magnitude in the order of 10^{-17} while in the first simulation it was in the order of 10^{0} and this is a result of the discussion above.

Chapter 7

CONCLUSIONS

We introduced a novel iterative blind equalization approach which is based on the iterative minimization of the l_∞ norm of the equalizer output using the subgradient optimization technique. SGBA updates are very simple and fast especially for DSP systems where the equalizer is implemented in hardware. As the computational requirement for the updates is minimal, the algorithm is expected to be insensitive against the round-off noise effects in fixed point implementations. Furthermore, due to the convex nature of the l_∞ cost function, the convergence problems that exist in CM type algorithms caused by the saddle and undesirable equilibria points and arbitrary initialization points do not exist. Further improvement in the convergence rate can be achieved through wise selection of the step-size, as illustrated by the examples in the previous section. Also the speed of convergence can be increased further by using weighting matrix proposed in chapter 4. Furthermore, the number of data samples consumed to blindly train the equalizer is less than the other iterative blind equalization algorithms such as CMA.

Moreover we have seen that the complexity of SGBA can be reduced and it can be made robust to noise by the use of a moving window. While the resulting algorithm keeps the reliability(guaranteed convergence, independence from the initial point) behavior of the standard SGBA there is a big decrease in the complexity. The algorithm still has incomparably high speed of convergence when compared to the CMA but now its complexity is lower then the CMA making it a better choice for systems requiring low complexity equalizers. Also this version makes tracking of a time varying channel possible which is another major improvement over the Fixed Window SGBA.

7.1 Future Work

We have seen that SGBA can be used for fractionally spaced channels as well as the symbol spaced channels. This is the result of our algorithm being an implicit HOS method. In Fractionally Spaced version of the algorithm, as stated in chapter 4, there is no need for any linear constraint which makes the algorithm even faster and makes the implementation easier. Fractionally spaced SGBA with weighting will be implemented as a future work. The convergence proof and the analysis of the convergence speed for this case will be made. The extension of SGBA to the MIMO case is another subject of interest. When the algorithm is extended to the MIMO case it will also be eligible for using in an arbitrary channel.
The algorithm can also be used in the fields of image restoration, PAR minimization, H^{∞} Equalization.
Finally it can be seen that the algorithm(and its versions) we proposed here has very high convergence speed and low complexity which makes it a great replacement for the current blind equalization algorithms whether it is a hardware only implementation or not.

BIBLIOGRAPHY

[1] D. Hatzinakos and C. L. Nikias, "Blind equalization using a trispectrum based algorithm," *IEEE Trans. on Comm.*, vol. 39, pp. 669–681, May 1991.

[2] B. Porat and B. Friedlander, "Blind equalization of digital communications channels using higher order moments," *IEEE Trans. ASSP*, vol. 39, pp. 522–526, Feb 1991.

[3] J. Tugnait, "Blind equalization and estimation of FIR communications channels using fractional sampling," *IEEE Trans. on Communications*, vol. 44, pp. 324–336, March 1996.

[4] D. N. Godard, "Self-recovering equalization and carrier tracking in two-dimensional data communication systems," *IEEE Trans. on Communications*, vol. 28, pp. 1867–1875, Nov. 1980.

[5] C. Johnson, J. Schniter, T. Endres, J. Behm, R. Casas, V. Brown, and C. Berg, "Blind equalization using the constant modulus criterion: A review," *Proc. of IEEE, Special Issue on Blind System Identificaiton and Estimation*, vol. 47, pp. 1927–1950, June 1998.

[6] H. Liu, G. Xu, L. Tong, and T. Kailath, "Recent developments in blind channel equalization: From cyclostationarity to subspaces," *Signal Processing*, vol. 50, pp. 83–99, April 1996.

[7] E. Moulines, P. Duhamel, J. Cardoso, and S. Mayrargue, "Subspace methods for the blind identification of multichannel FIR filters," *IEEE Trans. on SP*, vol. 43, pp. 516–525, Feb. 1995.

[8] W. Sethares, G. A. Rey, and C. Johnson, "Approaches to blind equalization of signals with multiple modulus," *Proceedings of ICASSP 98*, vol. 2, pp. 972–975, 1989.

[9] C. Papadias and D. Slock, "On the decision-directed equalization of constant modulus signals," *Proceedings of the Twenty-Eighth Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 1423–1427, 1994.

[10] T.-H. Li, "Finite-alphabet information and multivariate blind deconvolution and identification of linear systems," *IEEE Tran. on IT*, vol. 49, pp. 330–337, Jan. 2003.

[11] A. van der Veen, S. Talwar, and A. Paulraj, "Blind identification of FIR channels carrying multiple finite alphabet signals," *Proceedings of ICASSP 95*, vol. 2, pp. 1213–1216, 1995.

[12] L. Tong and S. Perreau, "Multichannel blind identification: From subspace to maximum likelihood methods," *Proceedings of the IEEE*, vol. 86, pp. 1951–1968, October 1998.

[13] S. Lambotharan, J. Chambers, and C. R. Johnson, "Attraction of saddles and slow convergence in CMA adaptation," *Signal Processing*, vol. 59, pp. 335–340, June 1997.

[14] S. Vembu, S. Verdu, R. Kennedy, and W. Sethares, "Convex cost functions in blind equalization," *IEEE Trans. on Signal Processing*, vol. 42, pp. 1952–1960, August 1994.

[15] D. P. Bertsekas, *Nonlinear Programming.* Athena Scientific, 1999.

[16] S. Boyd and L. Vandenberghe, *Convex Optimization.* Stanford University, 2001.

[17] J.-B. Hiriart-Urruty and C. Lemarechal, *Convex Analysis and Minimization Algorithms I.* Springer-Verlag, 1996.

[18] S. Haykin, *Adaptive Filter Theory.* Prentice Hall, 2002.

[19] A. H. Sayed, *Fundamentals of Adaptive Filtering.* John Wiley & Sons, 2003.

[20] B. T. Polyak, "A general method for solving extremal problems," *Doklady Akademii Nauk SSSR*, vol. 174, pp. 33–36, 1967.

[21] N. Shor, *Minimization Methods for Nondifferentiable Functions.* Springer-Verlag, 1985.

[22] A. Nedic and D. Bertsekas, "Convergence rate of incremental subgradient algorithms," *Stochastic Optimization: Algorithms and Applications, S.Ursayev and P.M. Pardalos, Editors*, 2000.

[23] U. Brannlund, *On Relaxation Methods for Nonsmooth Convex Optimization.* Ph.D. Thesis, Dept. of Mathematics, Royal Institute of Technology, Stockholm, 1993.

[24] J. Goffin, "On convergence rates of subgradient optimization methods," *Math. Programming*, vol. 13, pp. 329–347, May 1977.

[25] M. Held, P. Wolfe, and H. Crowder, "Validation of subgradient optimization," *Mathematical Programming*, vol. 6, pp. 62–88, February 1974.

[26] B. Polyak, "Minimization of unsmooth functionals," *USSR Computational Mathematics and Mathematical Physics*, vol. 9, pp. 509–521, 1969.

[27] E. Allen, R. Helgason, J. Kennington, and B. Shetty, "A generalization of polyak's convergence result for subgradient optimization," *Mathematical Programming*, vol. 37, pp. 309–317, May 1987.

[28] M. Bazaraa and H. Sherali, "On the choice of step size in subgradient optimization," *European Journal of OR.*, vol. 7, pp. 380–388, August 1981.

[29] S. Kim, H. Ahn, and S.-C. Cho, "Variable target value subgradient method," *Mathematical Programming*, vol. 49, pp. 359–369, January 1991.

[30] J. Goffin and K. Kiwiel, "Convergence of a simple subgradient level method," *Math. Programming*, vol. 85, no. 1, pp. 207–211, 1999.

[31] H. Sherali, G. Choi, and C. Tuncbilek, "Variable target value method for nondifferentiable optimization," *Operations Research Letters*, vol. 26, pp. 1–8, February 2000.

[32] O. Macchi, *Adaptive Processing: The Least Mean Squares Approach with Applications in Transmissions.* John Wiley & Sons, 1995.

[33] R. Schmidt, "Multiple emitter location and signal parameter estimation," *RADC Spectrum Estimation Workshop*, pp. 243 – 258, 1979.

[34] A. K. T. Roy, R.; Paulraj, "Esprit–a subspace rotation approach to estimation of parameters of cisoids in noise," *Acoustics, Speech, and Signal Processing*, pp. 1340 – 1342, 1986.

[35] O. Shalvi and E. Weinstein, "New criteria for blind deconvolution of nonminimum phase systems (channels)," *IEEE Trans. on Information Theory*, vol. 36, pp. 312–321, March 1990.

[36] ——, "Super-exponential methods for blind deconvolution," *IEEE Trans. on Information Theory*, vol. 39, pp. 504–519, March 1993.

[37] W. C. S. Todd K. Moon, *Mathematicial Tools for Signal Processing.*

[38] J. G. Proakis, *Digital Communications.* Mc-Graw Hill.

[39] I. M. M. K. M. W. F. I. Zhi-Quan (Tom) Luo, Member and J.-K. Zhang, "A fractionally spaced blind equalizer based on linear programming," *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, vol. 50, pp. 1650–1660, July 2002.

[40] D. A. Spielman and S.-H. Teng, "Smoothed analysis of termination of linear programming algorithms," *Mathematical Programming*, vol. 97, pp. 375–404, July 2003.

[41] C. Papadias, *Methods for Blind Equalization and Identification of Linear Channels.* Ph.D. Thesis, EUROCOM Institute, 1995.

[42] P. Regalia and M. Mboup, "Undermodeled equalization: a characterization of stationary points for a family of blind criteria," *IEEE Trans. on Signal Processing*, vol. 47, pp. 760–770, March 1999.

[43] ITU-T, *G.shdsl: Symmetric High-bitrate Digital Subscriber Line Standard*, 2001.

VITA

CAN KIZILKALE was born in Ankara, Turkey in 1981. He received his B.S. degree in department of Electrical and Electronics Engineering from Bilkent University, Ankara, Turkey in 2002. His research interests are adaptive blind equalization and convex optimization. Can Kizilkale is a student member of IEEE.