LOW COMPLEXITY CHANNEL ESTIMATION BASED ADAPTIVE LINEAR EQUALIZER

A Project Report

Submitted by

AKSHAY DHAVALE

In partial fulfilment of the requirements

for the award of the degree of

MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

May 2018

THESIS CERTIFICATE

This is to certify that the thesis titled, **LOW COMPLEXITY CHANNEL ESTIMATION BASED ADAPTIVE LIEANR EQUALIZER** submitted by **Akshay Dhavale**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Technology**, is a bona fide record of the research work done by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. K Giridhar
Project Guide
Professor
Dept. of Electrical Engineering
IIT-Madras, 600 036

Place: Chennai

Date: May 2018

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank my parents for their support and encouragement without which learning in and becoming a part of such a prestigious institution would not have been possible. I would like to dedicate this work to them.

I am ever thankful to Dr. K Giridhar for his continual support and invaluable advice throughout the duration of my project. I feel honoured and encouraged to have worked under his guidance. I would also like to take this opportunity to thank my professors at IIT Madras, Prof. Devendra Jalihal, and Prof.David Koilpillai who have imparted knowledge and have motivated me to learn the intricacies of the subjects. Also I am very much thankful to Madan for his help in my project.

ABSTRACT

KEYWORDS: **Channel Equalization, LLMSE, LMS, RLS**

Channel Equalization is a technique used at the receiver to combat intersymbol interference in time dispersive channels. The main causes of intersymbol interference are multipath propagation or non-linear frequency in channels. Intersymbol interference (ISI) is a signal distortion in telecommunication. One or more symbols can interfere with other symbols causing noise or a less reliable signal. Different kinds of Equalizer are available include Symbol spaced equalizer, Fractionally Spaced Equalizer, Blind Equalization, Decision-Feedback Equalization. An equalizer attempts to mitigate ISI and thus improve the receiver's performance but it is not possible to use a static equalizer because channel itself may vary with time. So a need for adaptive equalizer arises. Adaptive equalization determines the corrective filter in a dynamic manner based on the current channel Impulse Response. When using LLMSE (Linear Least Mean Squared Error Estimator) a need arises for inverse of covariance matrix which is expensive computationally. So we look for adaptation algorithms such as recursive least square (RLS), least mean square (LMS). RLS is still computationally expensive compared to LMS. But LMS has slower convergence rate than the RLS algorithm. The Proposed algorithm in this thesis has convergence rate similar to that of RLS and has lower complexity than LMS algorithm.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

ISI	Inter Symbol Interference
PDP	Power Delay Profile
LE	Linear Equalizer
LLMSE	Linear Least Mean Squared Error
DA LMS	Directly Adaptive Least Mean Square
DA RLS	Directly Adaptive Recursive Least Square
NLMS	Normalised Least Mean Square
DCD	Dichotomous Coordinate Descent
MSE	Mean Squared Error

NOTATION

$\boldsymbol{a}$	Bold face letters denote column vectors or matrices
$\widehat{\boldsymbol{b}}$	Estimate of the vector or a scaler
$\boldsymbol{h}$	Channel impulse response
M	Channel Estimator length
L	Equalizer vector length
l	Equalizer delay or decoding delay
n	Time instant
x	Transmitted symbols
y	Received signal
$\boldsymbol{e}_l$	Vector with all zeros except the lth element which is one
I_L	L x L Identity matrix
h_i	i th element of vector $\boldsymbol{h}$
$\boldsymbol{G}_{\boldsymbol{i},\boldsymbol{j}}$	$\boldsymbol{G}(i,j)$th element of matric G
$\Delta\boldsymbol{h}$	Represents change in the vector or matrix from $(i-1)$th to ith index
$\mathbf{b}_{\boldsymbol{i}:\boldsymbol{j}}$	Selecting ith to jth element of vector $\boldsymbol{b}$

CHAPTER 1

Introduction

Time dispersive wireless channels can cause intersymbol interference (ISI), that is, in a multipath scattering environment, the receiver sees delayed versions of a symbol transmission, which can interfere with other symbol transmissions Multipath Propagation, Bandlimited channels and fading are the major causes of ISI. An equalizer attempts to mitigate ISI and thus improve the system's performance. Because ISI channels are prevalent in communications systems, much effort has been devoted to developing effective equalizers for such channels.

Since the channel itself may be time varying need for adaptive equalization arises. There are distinct classes of adaptive equalizers such as linear adaptive equalizers and decision-feedback equalizers, each having a different overall configuration. They can use any one of the adaptive algorithms such as Least mean square (LMS), Signed LMS, Normalized LMS, Variable-step-size LMS, and Recursive least squares (RLS).

In this work we will consider linear Equalizers only. As discussed in the abstract when using Linear Least Mean Squared Error(LLMSE) Estimator need for the inversion of Autocovaraince matrix arises which is computationally expensive so we use Adaptive Algorithms such as Recursive Least Square (RLS),Least Mean Square(LMS) to find the equalization weight vector. LMS algorithm executes quickly but converges slowly, and its complexity grows linearly with the number of weights. The RLS algorithm converges quickly, but its complexity grows significantly. Also RLS has stability issues.

In this thesis we propose low complexity channel estimate based adaptive linear equalizer with complexity as low as $O(N_u(M+L))$, where M is the channel length and L is the Equalizer vector length and $N_u \ll M$ and $N_u \ll L$. Moreover using Dichotomous Coordinate Descent iterations for channel estimate and equalizer weight vector computation makes it multiplication free and division free which makes it attractive for hardware design. The proposed algorithm updates N_u taps of the channel estimator and N_u taps of the equalizer vector per received sample.

The results are obtained using MATLAB simulations. QPSK Modulated symbols are transmitted over the time dispersive channel.

For symbol spaced adaptive linear equalizer 2 types of channels considered are one is ped b channel with 9 taps and other is uniform Power Delay Profile(PDP) channel with 5 taps.

The results include scatter plots and Mean Squared Error (MSE) Learning Curve for LLMSE, Directly Adaptive RLS, Directly Adaptive NLMS and The Proposed Algorithm.

The key conclusion is that The Proposed Algorithm performs very close to the Directly Adaptive RLS and has complexity as low as that of LMS algorithm.

Flow of thesis:

The thesis is organized as follows.

Chapter 2 presents system model used for matlab simulations.This include system model for wireless channel , channel estimation and channel equalization.

Chapter 3 discusses adaptive linear equalizers.These include Linear Least Mean Square Error Equalizer, Directly Adaptive Least Mean Square and Directly Adaptive Recursive Least Square.

Chapter 4 discusses the Proposed Algorithm and how the computations are made simpler. It also discusses Dichotomous Coordinate Descant Algorithm.

Chapter 5 presents simulation results. These include MSE for channel estimation and Scatter Plots and Learning curves for symbol spaced linear equalizer for different adaptive algorithms.

Finally chapter 6 gives the conclusions and future work.

Chapter 2

System Model

2.1 Channel Model

An important characteristic of the wireless channel is the presence of many different paths between the transmitter and the receiver. Due to multipath propagation, more than one version of the transmitted signal arrive at the mobile receiver at slightly different times. The interference induced by these multiple copies, also known as multipath waves, has become the most significant cause of distortion known as fading and Inter-Symbol Interference (ISI).

There are a variety of ways to statistically model the wireless channels in order to represent the random behaviour of multipath fading. One simple and popular model represents the fading channel
with a linear and time-varying Channel Impulse Response (CIR) denoted by the function h (t,τ).

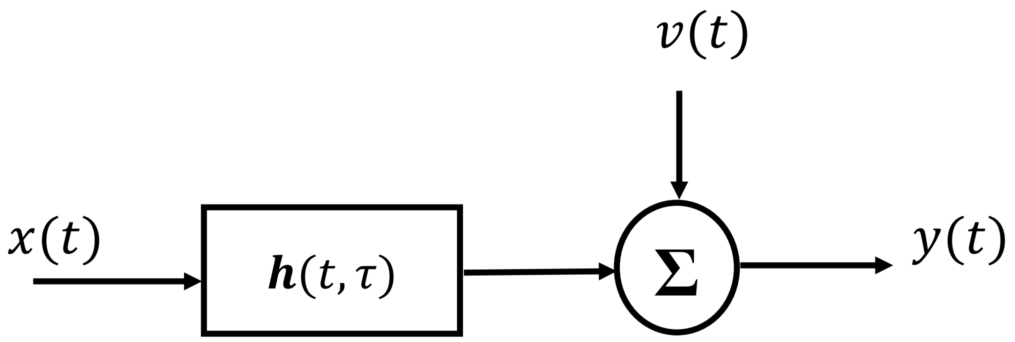

Figure 2.1: Continuous Time Channel Model

The Figure 2.1 shows continuous-time channel model of a wireless channel. $h\,(t,\tau\,)$ is the Channel Impulse Response (CIR) and $v(t)$ is the channel noise. The received signal $y\,(t)$ is given by

$$y\,(t) = \ h(t) * x(t) + v(t) \qquad (2.1)$$

where * represents convolution.

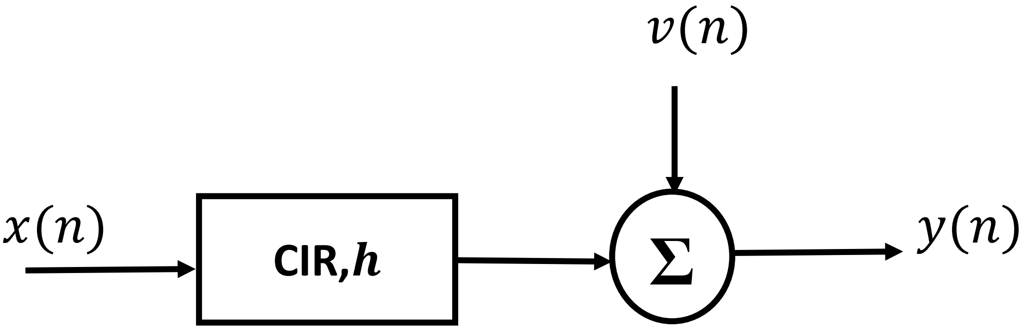

Figure 2.2 Discrete Time Channel Model

In digital communication system received signal $y(t)$ is sampled at $t = nT_s$, where T_s is the sampling period. Consequently, the Channel Impulse Response (CIR) can be written as a discrete-time impulse response $\boldsymbol{h} = (h_1, h_2, \dots, h_M)^T$, where M is the channel length. Furthermore, the transmitted and received signals are shown in discrete-time by their samples at the nth time step, $x[n] = x(nT_s)$ and $y[n] = y(nT_s)$.

Figure 2.2 shows discrete time channel model and received signal $y(n)$ is given by

$$y(n) = \boldsymbol{x}^T(n)\boldsymbol{h}(n) + v(n) \quad (2.2)$$

2.2 Channel Estimation

$$\begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ \vdots \\ \vdots \\ y(N) \end{bmatrix} = \begin{bmatrix} x(1) & 0 & \cdots & 0 \\ x(2) & x(1) & 0 & \vdots \\ \vdots & \vdots & \ddots & 0 \\ \vdots & \vdots & \ddots & x(1) \\ \vdots & \vdots & \ddots & \vdots \\ x(N) & x(N-1) & \cdots & x(N-M+1) \end{bmatrix} \boldsymbol{h} + \begin{bmatrix} v(1) \\ v(2) \\ \vdots \\ \vdots \\ \vdots \\ v(N) \end{bmatrix}$$

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{h} + \boldsymbol{v} \quad (2.3)$$

Where $y(1), y(2), \dots \dots, y(N)$ are received signal. $x(1), x(2), \dots \dots, x(N)$ are transmitted symbols. $\boldsymbol{h}$ is the unknown Channel Impulse response that we are trying to estimate and $\boldsymbol{v}$ is the additive white Gaussian noise with zero mean and variance σ_v^2.

For the above model from estimation theory least square estimate of the channel impulse response $\boldsymbol{h}$ is given by

$$\widehat{\boldsymbol{h}} = (\boldsymbol{H}^{\boldsymbol{H}}\boldsymbol{H})^{-\boldsymbol{1}}\boldsymbol{H}^{\boldsymbol{H}}\boldsymbol{y} \quad (2.4)$$

2.3 Channel Equalization

The Received signal at time instant n is given by

$$y(n) = \boldsymbol{x}^T(n)\boldsymbol{h}(n) + v(n) \quad (2.5)$$

Where

Transmitted symbol vector $\boldsymbol{x}(n) = [\, x(n)\; x(n-1) \ldots \ldots \ldots \ldots x(n-M+1)]^T$

Channel Impulse response $\boldsymbol{h}(n) = \; [h_1(n)\; h_2(n) \ldots \ldots \ldots \ldots h_M(n)]^T$

$v(n)$ is the white noise with zero mean and variance σ_v^2.

At time instant n , a L Length Linear Equalizer will estimate (n-l)th transmitted symbol as

$$\hat{\mathrm{x}}(\mathrm{n}-\mathrm{l}) = \boldsymbol{y}^T(n)\boldsymbol{f}(n) \quad (2.6)$$

Where $\boldsymbol{y}(n) \triangleq [y(n)\; y(n-1) \ldots \ldots \ldots y(n-L+1)]^T$ (2. 7)

And $\boldsymbol{f}(n)$ is the equalizer vector obtained by solving normal equations given below.

$$\boldsymbol{G}(n)\boldsymbol{f}(n) = \boldsymbol{\xi}(n) \quad (2.8)$$

Where

$$\boldsymbol{G}(n) = \overline{\boldsymbol{H}}^T(n)\overline{\boldsymbol{H}}(n) + \sigma_v^2\boldsymbol{I_L} \quad (2.9)$$

$$\boldsymbol{\xi}(n) = \overline{\boldsymbol{H}}^T(n)\boldsymbol{e}_l \quad (2.10)$$

$\boldsymbol{e}_l$ is $(L+M-1)$ x 1 vector of all zeros expect one at lth position.

$\overline{\boldsymbol{H}}(n)$ is the channel convolution matrix at time instant n. Since channel impulse response is unknown channel estimates $\widehat{\boldsymbol{h}}(n)$ are used to form $\overline{\boldsymbol{H}}(n)$ as follows.

$$\overline{\boldsymbol{H}}(n) = \begin{bmatrix} h_1(n) & 0 & \cdots & 0 \\ h_2(n) & h_1(n-1) & \cdots & \vdots \\ \vdots & h_2(n-1) & \ddots & \vdots \\ h_M(n) & \vdots & \ddots & \vdots \\ 0 & h_M(n-1) & \ddots & \vdots \\ \vdots & 0 & \ddots & 0 \\ \vdots & \vdots & \ddots & h_1(n-L+1) \\ \vdots & \vdots & \ddots & h_2(n-L+1) \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & h_M(n-L+1) \end{bmatrix} \quad (2.11)$$

Chapter 3

Adaptive Linear Equalizers

3.1 Introduction

Channel equalization is the process of reducing amplitude, frequency and phase distortion in a channel with the intent of improving transmission performance. This helps to remove errors produced due to inter symbol interference. Time-dispersive channels can cause inter-symbol interference (ISI). An equalizer attempts to mitigate ISI and thus improve the receiver's performance but here it is not possible to use a static equalizer. Because channels' impulse response vary with time. So a need for adaptive equalizer arises. Adaptive equalization determines the corrective filter in a dynamic manner based on the current channel transfer function. The same basic adaptive equalization principles (identification and correction) apply to both analog and digital communication systems. A model of the channel impulse response is determined based on information obtained from the transmitted signal, then a receiver filter that mitigates the channel distortion is synthesized. Many methods exist for both the identification and correction processes of adaptive equalization. The inter-symbol interference imposes the main obstacles to achieving increased digital transmission rates with the required accuracy. ISI problem is resolved by channel equalization. Frequently the channel parameters are not known in advance and moreover they may vary with time, in some applications significantly. Hence, it is necessary to use the adaptive equalizers, which provide the means of tracking the channel characteristics.

3.2 Symbol Spaced Linear Equalizer

A Symbol-Spaced linear equalizer consists of a tapped delay line that stores samples from the input signal. Once per symbol period, the equalizer outputs a weighted sum of the values in the delay line and updates the weights to prepare for the next symbol period. This class of equalizer is called Symbol-Spaced because the sample rates of the input and output are equal. In typical applications, the equalizer begins in training mode to gather information about the channel, and later switches to decision-directed mode.

Here, the new set of weights depends on these quantities:

a) The current set of weights

b) The input signal

c) The output signal

The advantage of symbol spaced equalizer is that it may offer lower complexity in some cases (not always). In most of the communication application we do not use symbol spaced equalizer because it is sensitive to timing phase.

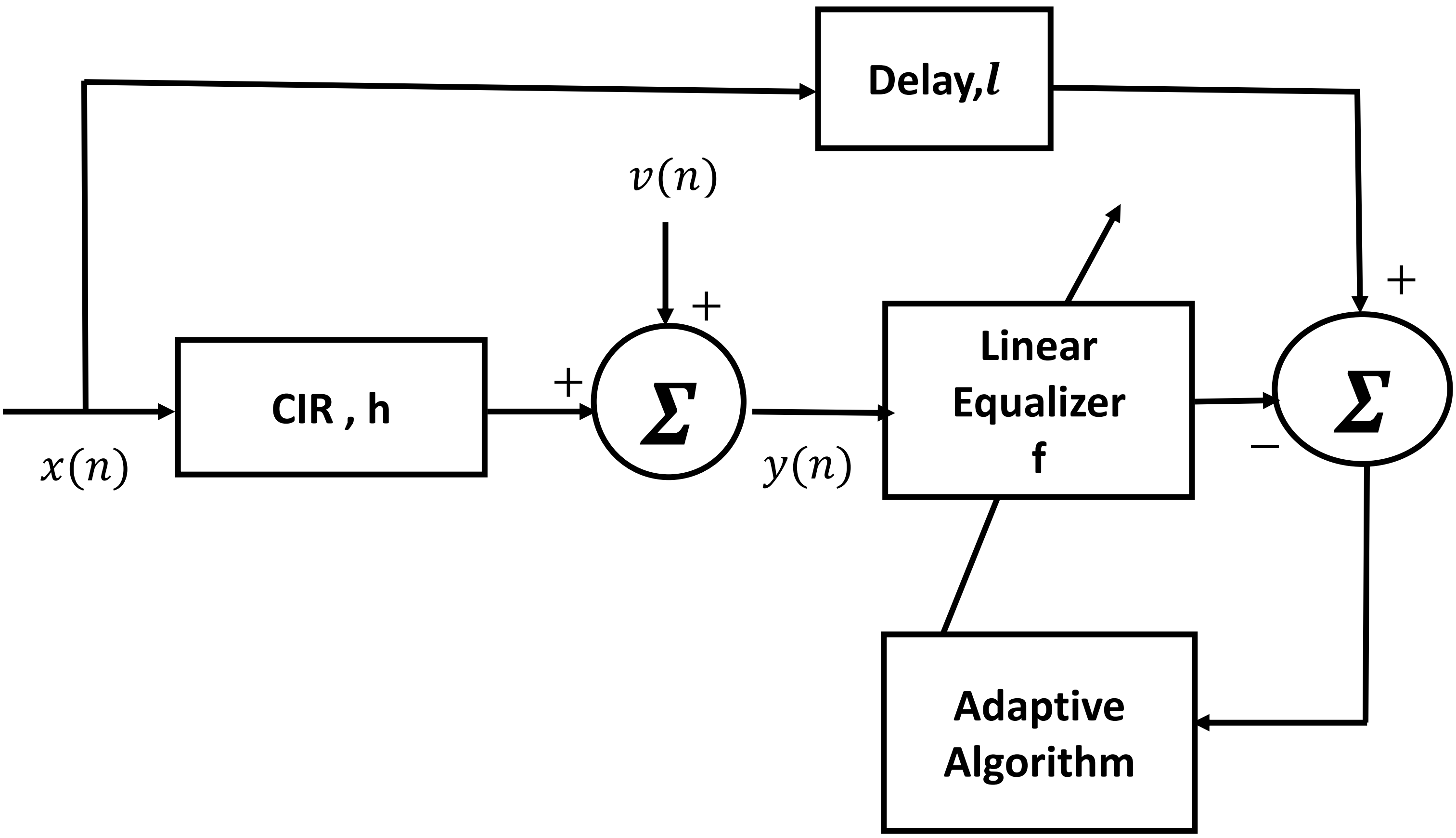

Figure 3.1: Adaptive Linear Equalizer Structure

3.3 Direct Adaptive Least Mean Square (DA LMS)

Least mean squares (LMS) algorithms are a class of adaptive filter used to mimic a desired filter by finding the filter coefficients that relate to producing the least mean squares of the error signal (difference between the desired and the actual signal). It is a stochastic gradient descent method in that the filter is only adapted based on the error at the current time. LMS filter is built around a transversal (i.e. tapped delay line) structure. Two practical features, simple to design, yet highly effective in performance have made it highly popular in various application. LMS filter employ, small step size statistical theory, which provides a fairly accurate description of the transient behaviour.

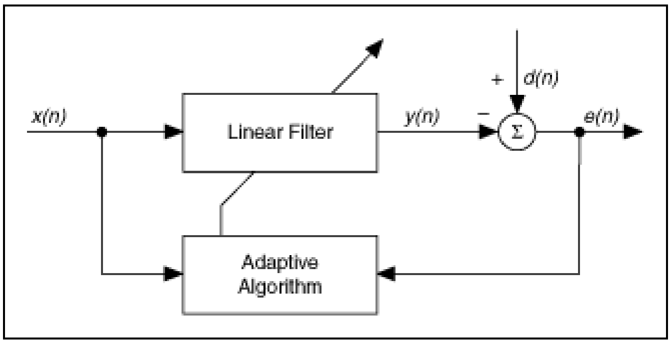

Figure 3.2: Adaptive Filter Structure

Figure 3.2 shows block diagram of an adaptive filter. Where

$x(n)$ =Input signal to linear filter at time instant n

$y(n)$ =Output signal to linear filter at time instant n

$d(n)$ =Desired signal at time instant n

$f(n)$ =Weight vector at time instant n

$e(n) = d(n) - f(n-1) * x(n)$ which is the a priory estimation error at time instant n

$e(n)$ is feedback to the adaptive algorithm which updates the weight vector according to the error.

LMS ITERATION

$$f(n) = f(n-1) + \mu\, x(n)^*\big(d(n) - f(n-1)\big)$$
$$= f(n-1) + \mu\, x(n)^*(e(n)) \qquad (3.1)$$

Where μ is the positive step size.

LMS algorithms executes quickly but converge slowly, and its complexity grows linearly with the no of weights. LMS is also known for its Computational simplicity.

NORMALISED LMS ITERATION

$$f(n) = f(n-1) + \frac{\mu}{(\varepsilon + ||x(n||^2)} x(n)^*(e(n)) \qquad (3.2)$$

$$f(0) = vector\ of\ Lx1\ zeros$$

Where μ is the positive step size and ε is small positive number.

In DA NLMS we will start with Equalization weight vector $f(0) = vector\ of\ Lx1\ zeros$ and keep updating the weight vector f according to the above update equation.

3.4 Direct Adaptive Recursive Least Square (DA RLS)

The Recursive least squares (RLS) adaptive filter is an algorithm which recursively finds the filter coefficients that minimize a weighted linear least squares cost function relating to the input signals. This in contrast to other algorithms such as the least mean squares (LMS) that aim to reduce the mean square error. In the derivation of the RLS, the input signals are considered deterministic, while for the LMS and similar algorithm they are considered stochastic. Compared to most of its competitors, the RLS exhibits extremely fast convergence. However, this benefit comes at the cost of high computational complexity, and potentially poor tracking performance when the filter to be estimated changes.

As illustrated in Figure 3.2, the RLS algorithm has the same to procedures as LMS algorithm, except that it provides a tracking rate sufficient for fast fading channel, moreover RLS algorithm is known to have the stability issues.

RLS ITERATION

$$P_i = \lambda^{-1}[P_{i-1} - \frac{\lambda^{-1} P_{i-1} x(n)^* x(n) P_{i-1}}{1 + \lambda^{-1} x(n) P_{i-1} x(n)^*} \qquad (3.3)$$

$$f(n) = f(n-1) + P_i x(n)^* \big(e(n)\big) \qquad (3.4)$$

Initial Conditions

$P_0 = \varepsilon I$ and $0 \ll \lambda \leq 1$

In simulation we will start with weight vector $f(0)$and keep updating the weight vector f according to the update equation 3.4.

Chapter 4

The Proposed Algorithm

This chapter describes in detail the proposed low complexity channel estimation based adaptive linear equalizer. The equalizer exploits the dichotomous coordinate descent (DCD) iterations for the estimate of the equalizer weight as well as channel estimate. The proposed technique has as low complexity as $O(Nu(L + M))$ operations per sample, where L and M are the equalizer and channel estimator length, respectively, and N_u is the number of iterations such that $N_u \ll L$ and $N_u \ll M$.The main idea used in this algorithm is that for every time instant n, channel Estimator is updated $N_u$times and also the equalizer vector is update N_u times. Because of this important assumption we get simplified recursions and we successfully avoid inverse of a matrix as well as matrix multiplications.

We start with the normal Equations

$$\mathbf{G}(\mathrm{n})\mathbf{f}(\mathrm{n}) = \boldsymbol{\xi}(\mathrm{n}) \qquad (4.1)$$

Where $\boldsymbol{G}(n) = \overline{\boldsymbol{H}}^T(n)\overline{\boldsymbol{H}}(n) + \sigma_v^2 \boldsymbol{I_L}$

$\boldsymbol{\xi}(n) = \overline{\boldsymbol{H}}^T(n)\boldsymbol{e}_l$

At every time instant n, $\boldsymbol{G}(n)$ and $\boldsymbol{\xi}(n)$ are known. But calculation of $\boldsymbol{G}(n)$ are expensive and so we come up with the below low complexity algorithm.

4.1 The Assumptions

Following are the assumptions

a) For every time sample n , the channel estimate $\widehat{\boldsymbol{h}}(n)$ can be up-dated N_u times. We will be using index $i = (n-1)N_u + k$, where k= 1,…..,N_u,to indicate such an update. Correspondingly, the sequence of the normal equations to be solved in the MMSE LE is now given by

$$\mathbf{G}(\mathrm{i})\mathbf{f}(\mathrm{i}) = \boldsymbol{\xi}(\mathrm{i}) \qquad (4.2)$$

b) For every i, the channel estimator updates only one, $p(i)$th ,element in $\widehat{\boldsymbol{h}}(i)$

$$\hat{\mathrm{h}}_{\mathrm{p(i)}}(\mathrm{i}) = \hat{\mathrm{h}}_{\mathrm{p(i)}}(\mathrm{i}-1) + \Delta\hat{\mathrm{h}}(\mathrm{i}) \qquad (4.3)$$

Where $\Delta\hat{h}(i)$ is the change in $p(i)th$ element of channel estimate.

c) For every i, only one $\mathrm{q}(i)$th, equalizer coefficient in $\hat{\boldsymbol{f}}(i)$ is updated as

$$\hat{\mathrm{f}}_{\mathrm{q(i)}}(\mathrm{i}) = \hat{\mathrm{f}}_{\mathrm{q(i)}}(\mathrm{i}-1) + \Delta\hat{\mathrm{f}}(\mathrm{i}) \tag{4.4}$$

Where $\Delta\hat{f}(i)$is the change in $q(i)th$ element of channel equalization vector.

d) Channel convolution matrix can be approximated for each i as

$$\boldsymbol{H}(i) = \begin{bmatrix} \hat{\boldsymbol{h}}(i) & 0 & \cdots & 0 \\ 0 & \hat{\boldsymbol{h}}(i) & \cdots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & \hat{\boldsymbol{h}}(i) \end{bmatrix}$$

4.2 Axilary Normal Equations

Normal equation for channel equalisation are given by

$$\mathbf{G}(\mathrm{i})\mathbf{f}(\mathrm{i}) = \boldsymbol{\xi}(\mathrm{i}) \tag{4.5}$$

We will convert above equation to auxiliary normal equation as follows.

At time instant $(i-1)$, $\boldsymbol{G}(i-1)\boldsymbol{f}(i-1) = \boldsymbol{\xi}(i-1)$ be approximately solved, and the let the approximate solution be $\hat{\boldsymbol{f}}(i-1)$.

Let residual vector be $\mathbf{r}(\mathrm{i}-1) = \boldsymbol{\xi}(\mathrm{i}-1) - \mathbf{G}(\mathrm{i}-1)\hat{\mathbf{f}}(\mathrm{i}-1)$ (4. 6)

At time instant (i), the above system is to be solved.

Denote $\Delta\boldsymbol{G}(i) = \boldsymbol{G}(i) - \boldsymbol{G}(i-1)$

$\Delta\boldsymbol{\xi}(i) = \boldsymbol{\xi}(i) - \boldsymbol{\xi}(i-1)$

$\Delta\boldsymbol{f}(i) = \boldsymbol{f}(i) - \hat{\boldsymbol{f}}(i-1)$

Our goal is to find $\hat{\boldsymbol{f}}(i)$ of system (4.5) by exploiting the previously obtained solution $\hat{\boldsymbol{f}}(i-1)$ and residual vector $\boldsymbol{r}(i-1)$.

Now $\boldsymbol{f}(i) = \hat{\boldsymbol{f}}(i-1) + \Delta\boldsymbol{f}(i)$

Expanding (4.5) as follows

$\boldsymbol{G}(i)\,[\hat{\boldsymbol{f}}(i-1)+\Delta\boldsymbol{f}(i)]=\boldsymbol{\xi}(i)$

$\boldsymbol{G}(i)\,\Delta\boldsymbol{f}(i)=\boldsymbol{\xi}(i)-\boldsymbol{G}(i)\hat{\boldsymbol{f}}(i-1)$

$=\boldsymbol{\xi}(i)-\boldsymbol{G}(i-1)\hat{\boldsymbol{f}}(i-1)-\Delta\boldsymbol{G}(i)\,\hat{\boldsymbol{f}}(i-1)$

$=\boldsymbol{r}(i-1)+\Delta\boldsymbol{\xi}(i)-\ \Delta\boldsymbol{G}(i)\,\hat{\boldsymbol{f}}(i-1)$

Auxiliary system of equations are,

$$\mathbf{G}(\mathrm{i})\ \Delta\mathbf{f}(\mathrm{i})=\boldsymbol{\xi}_{\mathbf{0}}(\mathrm{i}) \qquad (4.7)$$

Where $\boldsymbol{\xi}_{\mathbf{0}}(i)=\ \boldsymbol{r}(i-1)+\ \Delta\boldsymbol{\xi}(i)-\ \Delta\boldsymbol{G}(i)\,\hat{\boldsymbol{f}}(i-1)$

And obtain an approximate solution of the original system as $\hat{\boldsymbol{f}}(i)=\hat{\boldsymbol{f}}(i-1)+\Delta\hat{\boldsymbol{f}}(i)$ using DCD.

So at every index I we have equation (4.5).Next we will see how to solve a system of equations recursively.

4.2.1 Recursively Solving a system of equations

Table 4.1 shows steps in solving a system of equations recursively.

Residual Vector: $\boldsymbol{r}(i)=\boldsymbol{\xi}(i)-\boldsymbol{G}(i)\hat{\boldsymbol{f}}(i)$

$\Delta\boldsymbol{G}(i)=\boldsymbol{G}(i)-\boldsymbol{G}(i-1)$, $\Delta\boldsymbol{\xi}(i)=\boldsymbol{\xi}(i)-\boldsymbol{\xi}(i-1)$

Step	Equation
	Initialize: $\boldsymbol{r}(0)=0, \boldsymbol{\xi}(0)=0, \hat{\boldsymbol{f}}(0)=0$
1	Find $\Delta\boldsymbol{G}(i)$ and $\Delta\boldsymbol{\xi}(i)$
2	$\boldsymbol{\xi}_{\mathbf{0}}(i)=\boldsymbol{r}(i-1)+\Delta\boldsymbol{\xi}(i)-\Delta\boldsymbol{G}(i)\hat{\boldsymbol{f}}(i-1)$
3	Solve $\boldsymbol{G}(i)\Delta\boldsymbol{f}=\boldsymbol{\xi}_{\mathbf{0}}(i)\Rightarrow\ \Delta\hat{\boldsymbol{f}}(i), \boldsymbol{r}(i)$ using one iteration of DCD
4	$\hat{\boldsymbol{f}}(i)=\hat{\boldsymbol{f}}(i-1)+\Delta\hat{\boldsymbol{f}}(i)$

Table 4.1: Recursively Solving a System of Equations

4.3 Computation Of $\Delta\boldsymbol{G}(\boldsymbol{i})\hat{\boldsymbol{f}}(\boldsymbol{i}-\mathbf{1})$AND $\Delta\boldsymbol{G}(\boldsymbol{i})$

In the table 4.1, step1 requires finding $\Delta\boldsymbol{G}(i)$ which involves computation of the matrix

$\boldsymbol{G}(i) = \overline{\boldsymbol{H}}^T(i)\ \overline{\boldsymbol{H}}(i)$ with a complexity of $\mathcal{O}(M^2)$. Step2 requires $\mathcal{O}(ML)$ operations to compute

$\Delta\boldsymbol{G}(i)\hat{\boldsymbol{f}}(i-1)$. These are the most computationally demanding operations. In this section we show how

these two calculations are simplified using assumptions used from section 4.1.

4.3.1 Computation of $\Delta\boldsymbol{G}(\boldsymbol{i})\hat{\boldsymbol{f}}(\boldsymbol{i}-\mathbf{1})$

Let $\boldsymbol{H}(i) = \boldsymbol{H}(i-1) + \boldsymbol{\Delta}(i)$

Now $\Delta\boldsymbol{G}(i) = \overline{\boldsymbol{H}}^T(i)\ \overline{\boldsymbol{H}}(i) - \overline{\boldsymbol{H}}^T(i-1)\ \overline{\boldsymbol{H}}(i-1)$ (4.8)

$$\Delta\boldsymbol{G}(i)\hat{\boldsymbol{f}}(i-1) = \boldsymbol{\Delta}^T(i)\boldsymbol{H}(i-1)\hat{\boldsymbol{f}}(i-1) + \boldsymbol{H}^T(i-1)\boldsymbol{\Delta}(i)\hat{\boldsymbol{f}}(i-1) + \boldsymbol{\Delta}^T(i)\boldsymbol{\Delta}(i)\hat{\boldsymbol{f}}(i-1) \quad (4.9)$$

Now our Goal is to calculate $\Delta\boldsymbol{G}(i)\hat{\boldsymbol{f}}(i-1)$ in recursive fashion which involves three parts to simplify

i. $\boldsymbol{\Delta}^T(\mathrm{i})\mathbf{H}(\mathrm{i}-1)\hat{\boldsymbol{f}}(i-1)$

ii. $\boldsymbol{H}^T(i-1)\boldsymbol{\Delta}(i)\hat{\boldsymbol{f}}(i-1)$

iii. $\boldsymbol{\Delta}^T(i)\boldsymbol{\Delta}(i)\hat{\boldsymbol{f}}(i-1)$

i. $\boldsymbol{\Delta}^T(\mathrm{i})\mathbf{H}(\mathrm{i}-1)\hat{\boldsymbol{f}}(i-1)$

Denoting $\boldsymbol{b}(i-1) = \boldsymbol{H}(i-1)\hat{\boldsymbol{f}}(i-1)$

$$\boldsymbol{b}(i-1) = [\boldsymbol{H}(i-2) + \boldsymbol{\Delta}(i-1)]\left[\hat{\boldsymbol{f}}(i-2) + \Delta\hat{\boldsymbol{f}}(i-1)\right]$$

$$= \boldsymbol{b}(i-2) + \boldsymbol{H}(i-2)\Delta\hat{\boldsymbol{f}}(i-1) + \boldsymbol{\Delta}(i-1)\hat{\boldsymbol{f}}(i-1) \quad (4.10)$$

Consider term $\boldsymbol{H}(i-2)\Delta\hat{\boldsymbol{f}}(i-1)$ from the equation (4.9)

$$\boldsymbol{H}(i-2)\,\Delta\hat{\boldsymbol{f}}(i-1) = \begin{bmatrix} \hat{\boldsymbol{h}}(i-2) & 0 & \cdots & 0 \\ 0 & \hat{\boldsymbol{h}}(i-2) & \cdots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & \hat{\boldsymbol{h}}(i-2) \end{bmatrix} \begin{bmatrix} 0 \\ \vdots \\ \Delta\hat{f}(i-1) \\ 0 \\ \vdots \end{bmatrix}$$

$$= \Delta\hat{f}(i-1)\hat{\boldsymbol{h}}^{[q(i-1)]}(i-2) \tag{4.11}$$

Now consider last term from the equation (4.9)

$$\boldsymbol{\Delta}(i-1)\hat{\boldsymbol{f}}(i-1) = \begin{bmatrix} \mathbf{0}_{\boldsymbol{p(i-1)-1}} & \mathbf{0} & \cdots & \mathbf{0} \\ \Delta\hat{h}(i-1) & \mathbf{0}_{\boldsymbol{p(i-1)-1}} & \cdots & \vdots \\ \mathbf{0} & \Delta\hat{h}(i-1) & \ddots & \mathbf{0}_{\boldsymbol{p(i-1)-1}} \\ \vdots & \mathbf{0} & \ddots & \Delta\hat{h}(\boldsymbol{i-1}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{f}}_{\boldsymbol{1}}(i-1) \\ \hat{\boldsymbol{f}}_{\boldsymbol{2}}(i-1) \\ \vdots \\ \vdots \\ \hat{\boldsymbol{f}}_{\boldsymbol{L}}(i-1) \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \hat{\boldsymbol{f}}_{\boldsymbol{1}}(i-1) \\ \vdots \\ \hat{\boldsymbol{f}}_{\boldsymbol{L}}(i-1) \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}$$

$$= \boldsymbol{\Delta}\hat{\boldsymbol{h}}(i-1)\hat{\boldsymbol{f}}^{[\boldsymbol{p(i-1)}]}(i-1) \tag{4.12}$$

So recursion for b now becomes

$$\boldsymbol{b}(i-1) = \boldsymbol{b}(i-2) + \Delta\hat{\boldsymbol{f}}(i-1)\hat{\boldsymbol{h}}^{[\boldsymbol{q}(i-1)]}(i-2) + \boldsymbol{\Delta}\hat{\boldsymbol{h}}(i-1)\hat{\boldsymbol{f}}^{[\boldsymbol{p}(i-1)]}(i-1) \tag{4.13}$$

Now coming back to first term in $\Delta\boldsymbol{G}(i)\hat{\boldsymbol{f}}(i-1)$

$$\boldsymbol{\Delta}^{\boldsymbol{T}}(i)\boldsymbol{H}(i-1)\hat{\boldsymbol{f}}(i-1) = \boldsymbol{\Delta}^{\boldsymbol{T}}(i)\boldsymbol{b}(i-1)$$

$$= \begin{bmatrix} \mathbf{0}_{\boldsymbol{p(i)-1}} & \mathbf{0} & \cdots & \mathbf{0} \\ \Delta\hat{\boldsymbol{h}}(i) & \mathbf{0}_{\boldsymbol{p(i)-1}} & \cdots & \vdots \\ \mathbf{0} & \Delta\hat{\boldsymbol{h}}(i) & \ddots & \mathbf{0}_{\boldsymbol{p(i)-1}} \\ \vdots & \mathbf{0} & \ddots & \Delta\hat{\boldsymbol{h}}(i) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}^{\boldsymbol{T}} \boldsymbol{b}(i-1) = \begin{bmatrix} \mathbf{0}_{\boldsymbol{p(i)-1}} & \Delta\hat{\boldsymbol{h}}(i) & \mathbf{0} & \cdots & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{0}_{\boldsymbol{p(i)-1}} & \Delta\hat{\boldsymbol{h}}(i) & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0}_{\boldsymbol{p(i)-1}} & \Delta\hat{\boldsymbol{h}}(\boldsymbol{i}) \end{bmatrix} \boldsymbol{b}(i-1)$$

Finally the first part becomes

$$\boldsymbol{\Delta}^{\boldsymbol{T}}(i)\boldsymbol{H}(i-1)\hat{\boldsymbol{f}}(i-1) = \Delta\hat{\boldsymbol{h}}(\boldsymbol{i})\ \boldsymbol{b}_{\boldsymbol{p(i):p(i)+L-1}}(i-1) \tag{4.14}$$

ii. $\boldsymbol{H}^{T}(i-1)\boldsymbol{\Delta}(i)\hat{\boldsymbol{f}}(i-1)$

This part of recursion is figured out by Madan and I am writing the final result here for the sake of completeness.

Similar to the first part we get following simplified form for second part

$$\boldsymbol{H}^{T}(i-1)\boldsymbol{\Delta}(i)\hat{\boldsymbol{f}}(i-1) = \Delta\hat{h}(i)\boldsymbol{c}_{M-p(i)+1:M-p(i)+L}(i-1) \tag{4.15}$$

4.3.2 Computation of $\Delta \boldsymbol{G}(\boldsymbol{i})$

Now since $\boldsymbol{G}(i)$, $\Delta\boldsymbol{G}(i)$ are Symmetric Toeplitz matrices for each update of the channel estimate, only the first column of $\Delta\mathrm{G}(\mathrm{i})$ needs to be updated.

$$\Delta\mathrm{G}(\mathrm{i}) = \overline{\boldsymbol{H}}^T(i)\,\overline{\boldsymbol{H}}(i) - \overline{\boldsymbol{H}}^T(i-1)\,\overline{\boldsymbol{H}}(i-1)$$

$$\overline{\boldsymbol{H}}^T(i)\,\overline{\boldsymbol{H}}(i) = \begin{bmatrix} \hat{h}_1(i) & \cdots & \hat{h}_M(i) & 0 & \cdots & 0 \\ 0 & \hat{h}_1(i) & \cdots & \hat{h}_M(i) & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix} \begin{bmatrix} \hat{h}_1(i) & 0 & \cdots \\ \vdots & \hat{h}_1(i) & \ddots \\ \hat{h}_M(i) & \vdots & \ddots \\ 0 & \hat{h}_M(i) & \ddots \\ \vdots & \vdots & \ddots \\ 0 & 0 & \ddots \end{bmatrix}$$

$$\Delta\boldsymbol{G}_{1,1}(i) = \sum_{l=1}^{M} \hat{h}_l^2(i) - \hat{h}_l^2(i-1) = \hat{h}_{p(i)}^2(i) - \hat{h}_{p(i)}^2(i-1) = \Delta\hat{h}(i)\left[\hat{h}_{p(i)}(i) + \hat{h}_{p(i)}(i-1)\right] \quad (4.16)$$

$$\Delta\boldsymbol{G}_{1,m}(i) = \sum_{l=m}^{M} \hat{h}_l(i)\,\hat{h}_{l-m+1}(i) - \hat{h}_l(i-1)\,\hat{h}_{l-m+1}(i-1) \quad (4.17)$$

$$= \hat{h}_{p(i)-m+1}(i-1)\left[\hat{h}_{p(i)}(i) - \hat{h}_{p(i)}(i-1)\right] + \hat{h}_{p(i)+m-1}(i-1)\left[\hat{h}_{p(i)}(i) - \hat{h}_{p(i)}(i-1)\right]$$

$= \Delta\hat{h}(i)\left[\hat{h}_{p(i)-m+1}(i-1) + \hat{h}_{p(i)+m-1}(i-1)\right]$ for $m = 2,3 \ldots . M$ and $m < p(i) < M - m + 1$

4.4 The Proposed Algorithm

The Above proposed channel estimate based low complexity adaptive equalization algorithm is summarized in table 4.2.

Step	Equation
	Initialization: $i=0, \boldsymbol{G}(0)=\sigma_v^2\boldsymbol{I_L}, \hat{\boldsymbol{f}}(\boldsymbol{0})=\boldsymbol{0}, \boldsymbol{r}(\boldsymbol{0})=\boldsymbol{0}, \boldsymbol{b}(\boldsymbol{0})=\boldsymbol{0}, \boldsymbol{c}(\boldsymbol{0})=\boldsymbol{0}$
	For $n=1,2, \dots\dots\dots$
	For $k=1, \dots, N_u$
1	$i=i+1$
2	*Obtain* $\widehat{\boldsymbol{h}}(i)$, $\Delta\hat{h}(i-1)$ *and postion* $p(i)$ *from channel estimator.*
3	$\boldsymbol{z}(i)=\boldsymbol{\Delta}\hat{h}(i)[\boldsymbol{b}_{\boldsymbol{p}(\boldsymbol{i}):\boldsymbol{p}(\boldsymbol{i})+\boldsymbol{L}-\boldsymbol{1}}(i-1)+\boldsymbol{c}_{\boldsymbol{M}-\boldsymbol{p}(\boldsymbol{i})+\boldsymbol{1}:\boldsymbol{M}-\boldsymbol{p}(\boldsymbol{i})+\boldsymbol{L}}(\boldsymbol{i}-\boldsymbol{1})+\Delta\hat{h}(\boldsymbol{i})\hat{\boldsymbol{f}}(i-1)$
4	$\boldsymbol{\xi_0}(i)=\boldsymbol{r}(i-1)+\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(i)\boldsymbol{e}_{\boldsymbol{l}+\boldsymbol{p}(i)}-\boldsymbol{z}(i)$
5	Compute $\Delta G^{(1)}(i)$ and update $\boldsymbol{G}^{(1)}(i)=\boldsymbol{G}^{(1)}(i-1)+\Delta\boldsymbol{G}^{(\boldsymbol{1})}(i)$
6	Use one iteration to solve $\boldsymbol{G}(i)\ \Delta\boldsymbol{f}(i)=\boldsymbol{\xi_0}(i)$ and obtain $\Delta\hat{f}(i), q(i)$ and $\boldsymbol{r}(i)$
7	$\hat{f}_{q(i)}(i)=\hat{f}_{q(i)}(i-1)+\Delta\hat{f}(i)$
8	$\boldsymbol{b}(i-1)=\boldsymbol{b}(i-2)+\Delta\hat{\boldsymbol{f}}(i-1)\hat{h}^{[q(i-1)]}(i-2)+\Delta\hat{h}(i-1)\hat{\boldsymbol{f}}^{[p(i-1)]}(i-1)$
9	$\boldsymbol{c}(i-1)=\boldsymbol{c}(i-2)+\Delta\hat{f}(i-1)\widehat{\boldsymbol{u}}^{[q(i-1)]}(i-2)+\Delta\hat{h}(i-1)\hat{\boldsymbol{f}}^{[M-p(i-1)+1]}(i-1)$

Table 4.2 The Proposed Algorithm

4.5 Dichotomous Coordinate Descent Algorithm

We propose to use the DCD iteration described in Table 4.3, which is simple for implementation and shows fast convergence to optimal performance. When using the DCD iteration, it is assumed that the equalizer coefficients are represented as M_b-bit fixed-point numbers within an interval [−A,A], where A is preferably a power-of-two number. The step-size parameter α is α = $2^{-a}A$, i.e. also a power-of two number. With such settings, operations required in the DCD algorithm are only additions as all multiplications and divisions are replaced by bit-shifts.

If, in addition, the adaptive channel estimator is implemented using the RLS-DCD adaptive filter of complexity (N_uM) , the increments $\Delta\hat{h}(i)$ will be power of two numbers. Therefore, all multiplications in Table 4.2 can be replaced by bit-shift operations.

DCD algorithm with one update is summarized in the table 4.3.

Step	Equation
	Initialization: $\boldsymbol{r} = \boldsymbol{\xi_0}$, $\alpha = \frac{A}{2}, a = 1$
1	$q = argmax_{j=1,\ldots,L}\{\lvert r_j \rvert\}$, Go to Step 4
2	$a = a + 1; \alpha = \alpha/2$
3	If $a > M_b$ the algorithm stops
4	If $\lvert r_q \rvert < G_{q,q}$ then go to step 2
5	$\Delta\hat{f} = sign(r_q)\ \alpha$
6	$r = r - sign(r_q)\ \alpha\ G^{(q)}$
	$\Delta\hat{f}(i) = \Delta\hat{f}$, $q(i) = q, \boldsymbol{r}(i) = \boldsymbol{r}$

Table 4.3 DCD with one update

CHAPTER 5

Simulation and Results

The results are obtained using MATLAB simulations. QPSK modulation is used to simulate the system. Two kind of channel models are used one is uniform PDP channel with 5 taps and other is ped b channel of 9 taps.

5.1 Channel Estimation MSE Learning Curves

We compare learning curves of optimal channel estimator with the DCD based channel estimation at SNR =15dB.Channel used is uniform Power Delay Profile(PDP) of 5 taps. Estimation error is averaged over 500 channel realizations.

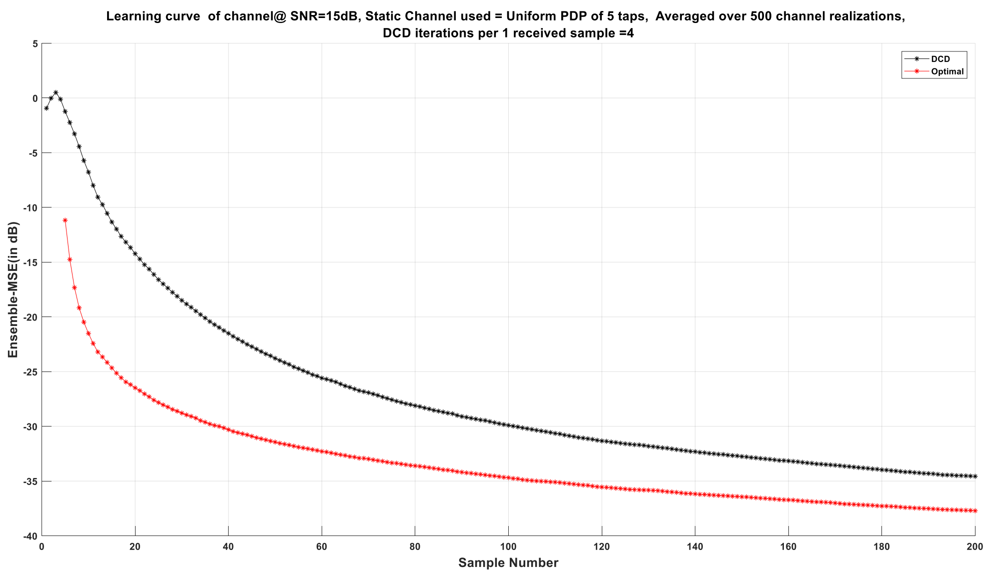

Figure 5.1: Channel Estimation MSE Optimal vs DCD

5.2 Scatter Plots For Channel Equalization

Scatter plots of LLMSE,Directly Adpative RLS,Directly adaptive NLMS and the Proposed algorithm at SNR=25dB are presented in this section. Channel used is Ped b channel with 9 taps.

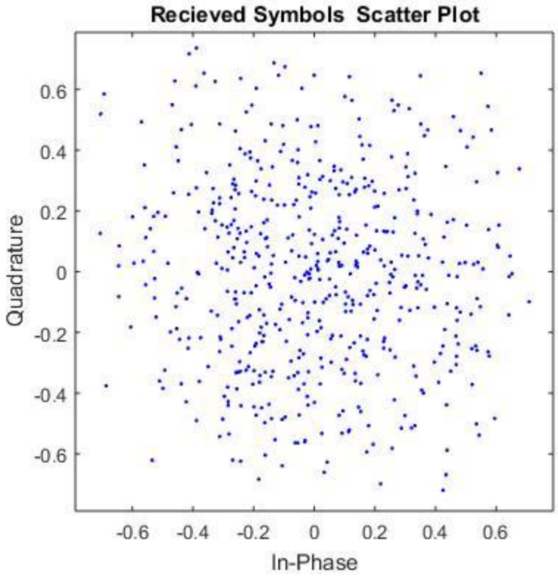

Figure 5.2: Received Symbols

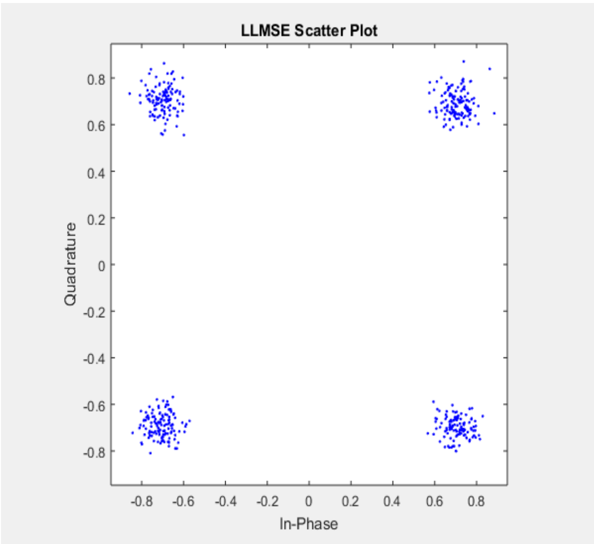

Figure 5.3: LLMSE Scatter Plot

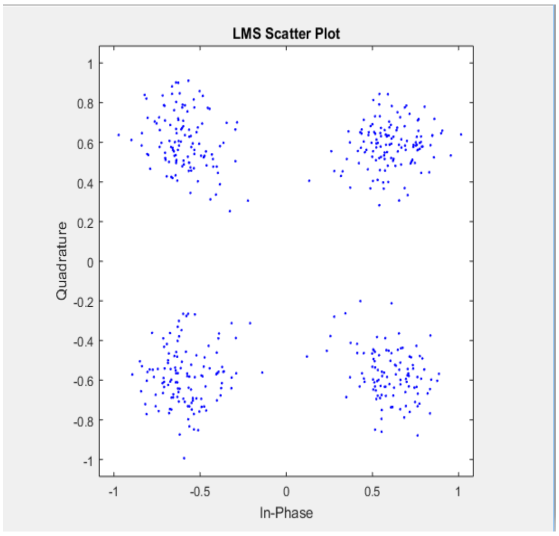

Figure. 5.4: DA LMS Scatter Plot

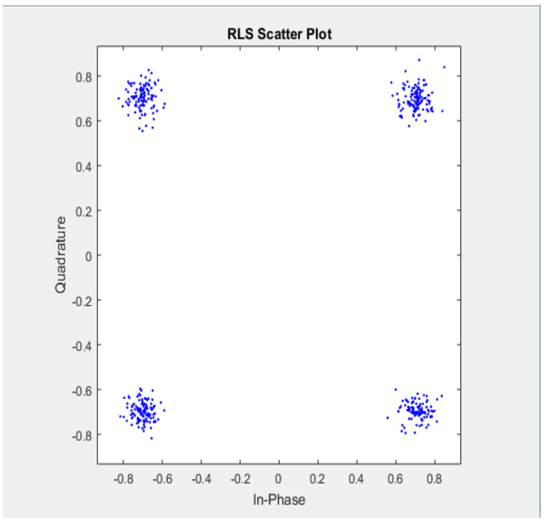

Figure 5.5: DA RLS scatter plot

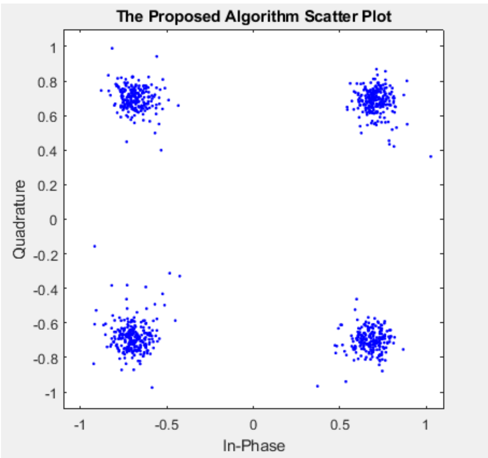

Figure5.6: The Proposed Algorithm Scatter Plot

5.3 Channel Equalization MSE Learning Curves

Learning Curves for symbol spaced channel equalization of LLMSE, Directly Adaptive RLS, Directly adaptive NLMS and the proposed algorithm are presented in this section.

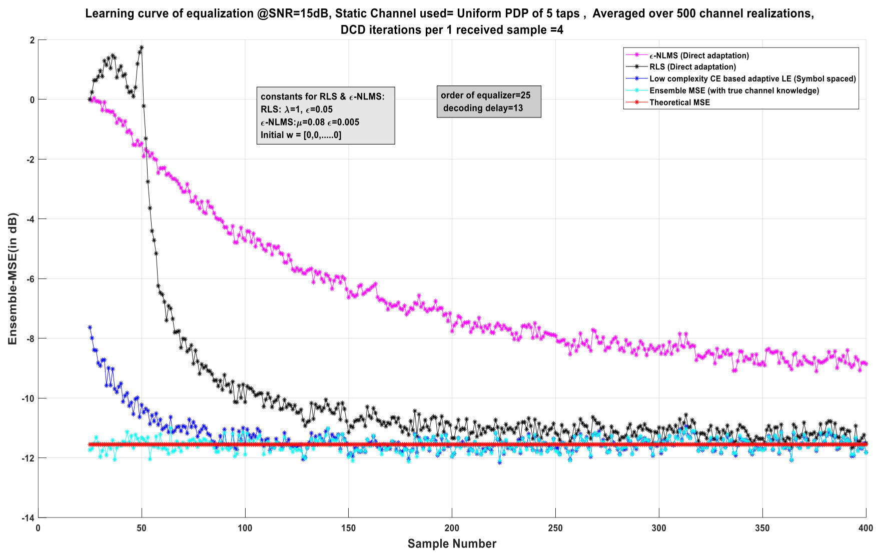

Figure 5.7 Learning curves for Uniform PDP channel of 5 taps

Figure 5.7 shows learning curves of channel equalization for static uniform PDP channel of 5 taps at SNR =15dB . Error is averaged over 500 channel realizations. We can observe that proposed algorithm performs very closely with the LLMSE.

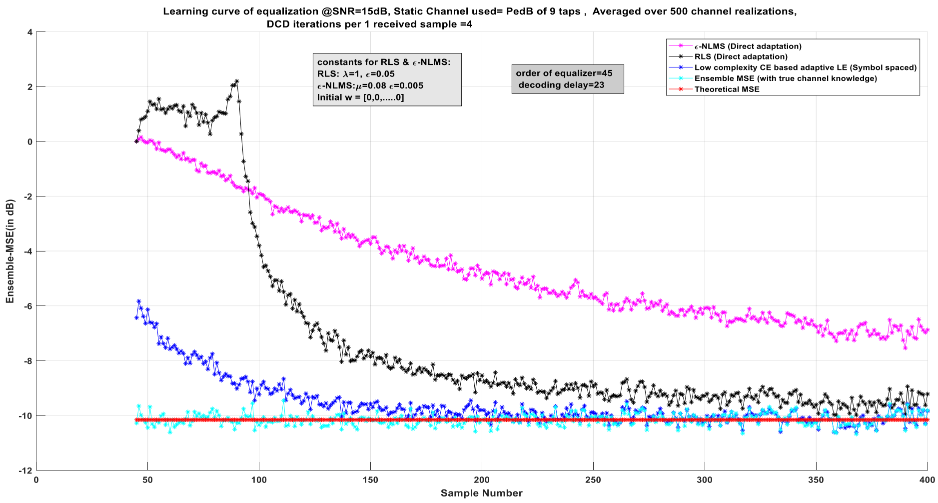

Figure 5.8 Learning curves for Ped b channel of 9 taps

Figure 5.8 shows learning curves of channel equalization for static Ped b channel of 9 taps at SNR =15dB . Error is averaged over 500 channel realizations. We can observe that proposed algorithm performs very closely with the LLMSE.

CHAPTER 6

Conclusions

In this thesis we presented the low complexity channel estimate based adaptive linear equalizer and compared learning curves for the symbol spaced linear equalizer of the LLMSE ,DA RLS , DA NLMS algorithms and the proposed algorithm. So we conclude that the proposed low complexity channel estimation based adaptive linear algorithm performs very close to the Linear Least mean Square Estimator which is the optimal estimator. Moreover, when using the dichotomous coordinate descent iterations, computation of the equalizer coefficients is multiplication-free and division free, which makes it attractive for hardware design.

Future work includes extending the proposed algorithm to the fractionally spaced linear equalizer.

References

[1] Ali H Sayad, Adaptive Filters Wiley, 2008

[2] J. Proakis, Digital communications

[3] Chen, T., Zakharov, Y. V., & Liu, C. (2011). Low-complexity channel-estimate based adaptive linear equalizer. *IEEE Signal Processing Letters*, *18*(7), 427-430

[4] Y. Zakharov, G. White, and J. Liu, "Low-complexity RLS algorithms using dichotomous coordinate descent iterations," IEEE Transactions onSignal Processing, vol. 56, no. 7 Part 2,pp. 3150–3161, 2008.