CHAPTER 1 – INTRODUCTION

1.1 A brief background of wireless communications.

When land line based telephony systems were first introduced in the earlier part of this century, they allowed people to communicate almost instantaneously over large distances at a reasonable cost for the very first time. Indeed it was the advent of telephony and the advance in communications technology that has drastically influenced the way we live and our outlook on life.

If land based telephony ushered in the age of communications; then wireless communications is its legitimate successor. Though land based systems will go on providing backbone connectivity for a long time to come, it has become increasingly clear in the last decade or so, that for the last hop in the information chain (that links the user to his information source), the user prefers wireless access. The reason for this is simply convenience. Nobody wants his or her mobility restricted. Wireless access allows the user the freedom to be mobile. Apart from user satisfaction, there is a very legitimate justification of wireless connectivity from the service provider's point of view. There are many areas in the world that are still inaccessible to land line systems due to their remoteness or because of intervening inhospitable terrain. Wireless systems are a very practical alternate in such a scenario to replace or supplement the backbone landlines.

The last and perhaps most important factor in the drive towards mobile telephony is simply economics *[1]*. For the first 35 years since its first commercial deployment ,wireless systems saw little market penetration due to the high cost and the technological challenges involved. But in the last fifteen years, cellular telephony alone (not including paging, amateur radio, terrestrial microwave radio systems) has been growing at rates similar to that of television and the automobile as seen in the

figure below. So we see that wireless communications holds enough promise to be the technology that drives our lifestyle and indeed our culture into the next millenium.

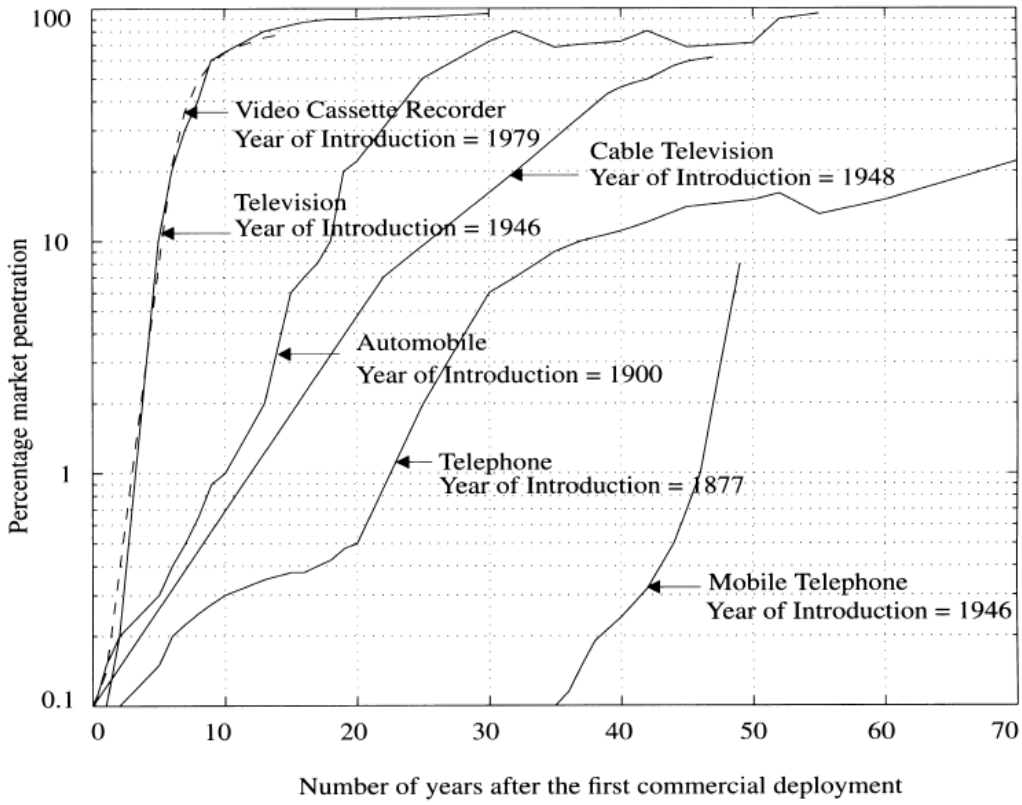


Fig 1.1: Mobile Telephony growth compared with other inventions this century

As with most things this promising, wireless communications opens a whole Pandora's box of complications. A radio signal transmitted from a base station to a mobile in a typical urban environment, exhibits variations in both received amplitude and phase/frequency. The change in amplitude us usually manifest in the form of sharp drops in signal level called *Fades*. Fades of 40dB or more below the mean signal level are common with successive minima occurring every few inches of the motion of the mobile. A vehicle traveling at 60 miles/hr can easily experience fades at the rate of 100Hz, thus distorting speech *[1]*.Each radio wave received at the mobile has an associated Doppler shift that depends on the mobile velocity, the carrier frequency and

the angle between the velocity vector and the wave propagation vector. This manifests itself as a random variance in the instantaneous frequency of the received signal, causing further distortion.

These obstacles may seem to defy any attempt at a systematic interpretation or analysis. However, starting from a model that takes into account the buildings and other structures in the vicinity of the mobile, that affect the signal, we can successfully predict many of the observed properties of the received signal by utilizing statistical techniques.

1.2 What is Channel Estimation?

Before we approach the problem of predicting and analyzing the observable properties of transmission, we must first define what we mean by a channel. In its most general sense, a *channel* can describe everything from the source to the sink of a radio signal *[3]*. This includes the physical medium (free space, fiber, waveguides etc.) between the transmitter and the receiver through which the signal propagates. The word channel refers to this physical medium throughout this work. An essential feature of any physical medium is, that the transmitted signal is received at the receiver, corrupted in a variety of ways by frequency and phase-distortion, inter symbol interference and thermal noise.

A *channel model* on the other hand can be thought of as a mathematical representation of the transfer characteristics of this physical medium. This model could be based on some known underlying physical phenomenon or it could be formed by fitting the best mathematical / statistical model on the observed channel behavior. Most channel models are formulated by observing the characteristics of the received signals for each specific environment. Different mathematical models that explain the received signal are then fit over the accumulated data. Usually the one that best explains the behavior of the received signal is used to model the given physical channel.

Channel estimation[7] is simply defined as the process of characterizing the effect of the physical channel on the input sequence. If the channel is assumed to be linear, the channel estimate is simply the estimate of the impulse response of the system. It must be stressed once more that channel estimation is only a mathematical representation of what is truly happening. A "good" channel estimate is one where some sort of error minimization criteria is satisfied (e.g. MMSE).

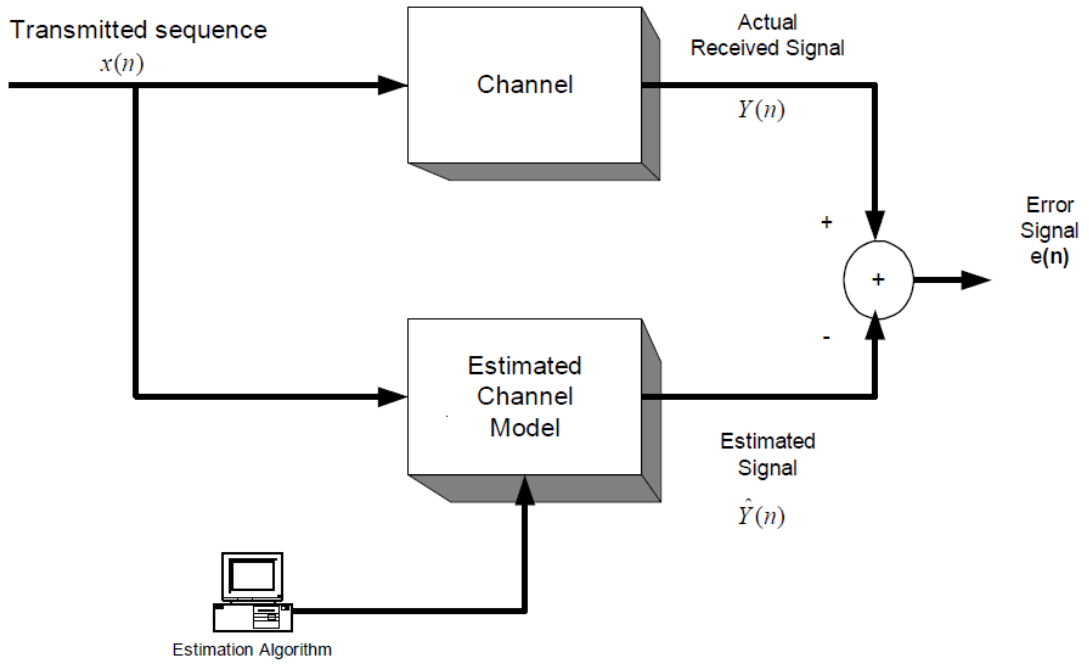


Fig: 1.2: A general Channel Estimation Procedure.

In the figure above *e(n)* is the estimation error. The aim of most channel estimation algorithms is to minimize the mean squared error (MMSE), $E[e^2(n)]$ while utilizing as little computational resources as possible in the estimation process.

1.3 Why Channel Estimation?

Channel estimation algorithms allow the receiver to approximate the impulse response of the channel and explain the behavior of the channel. This knowledge of the

channel's behavior is well-utilized in modern radio communications. Adaptive channel equalizers utilize channel estimates to overcome the effects of inter symbol interference. Diversity techniques (for e.g. the IS-95 Rake receiver) utilize the channel estimate to implement a matched filter such that the receiver is optimally matched to the received signal instead of the transmitted one. Maximum likelihood detectors utilize channel estimates to minimize the error probability.

One of the most important benefits of channel estimation is that it allows the implementation of coherent demodulation*[4]*. Coherent demodulation requires the knowledge the phase of the signal. This can be accomplished by using channel estimation techniques.

1.4 Filters for Channel Equalization

In order to counter intersymbol interference effect, the observed signal may first be passed through a filter called the equalizer whose characteristics are the inverse of the channel characteristics. If the equalizer is exactly matched to the channel, the combination of the channel and equalizer is just a gain so that there is no intersymbol interference present at the output of the equalizer. As mentioned, the equalizer is a filter which is known as Adaptive filter.

1.4.1. Adaptive Filter

In contrast to filter design techniques based on knowledge of the second-order statistics of the signals, there are many digital signal processing applications in which these statistics cannot be specified a priori. The filter coefficients depend on the characteristics of the medium and cannot be specified a priori. Instead, they are determined by the method of Least squares, from measurements obtained by transmitting signals through the physical media. Such filters, with adjustable parameters, are usually called adaptive filters, especially when they incorporate

algorithms that allow the filter coefficients to adapt to the changes in the signal statistics.

The equalizers, thereby using adaptive filters are called adaptive equalizers. On channels whose frequency response characteristics are unknown, but time invariant, we may measure the channel characteristics and adjust the parameters of the equalizer; once adjusted, the parameters remain fixed during the transmission of data. Such equalizers are called preset equalizers. On the other hand, adaptive equalizers update their parameters on a periodic basis during the transmission of the data and, thus, they are capable of tracking time-varying channel response.

The adaptive filters will be discussed, in detail, in the next chapter. However, at this point of time, one needs to understand that the equalizer used to counter intersymbol interference effect of the channel is to be adaptive in nature. This is because of the reason that, there is no priori information available to the filter but only the incoming data, depending on which the filter parameters have to adapt.

1.5 Training Sequences vs. Blind Methods

Once a model has been established, its parameters need to be continuously updated (estimated) in order to minimize the error as the channel changes. If the receiver has a-priori knowledge of the information being sent over the channel, it can utilize this knowledge to obtain an accurate estimate of the impulse response of the channel. This method is simply called *Training sequence based Channel estimation*. It has the advantage of being used in any radio communications system quite easily. Even though this is the most popular method in use today, it still has its drawbacks. One of the obvious drawbacks is that it is wasteful of bandwidth. Precious bits in a frame that might have been otherwise used to transport information are stuffed with training sequences for channel estimation. This method also suffers due to the fact that most

communication systems send information lumped frames. It is only after the receipt of the whole frame that the channel estimate can be extracted from the embedded training sequence. For fast fading channels this might not be adequate since the coherence time of the channel might be shorter than the frame time.

*Blind methods [4,5]*on the other hand require no training sequences. They utilize certain underlying mathematical information about the kind of data being transmitted. Classical equalization techniques employ a time-slot (recurring periodically for time- varying situations) during which a training signal, known in advance by the receiver, is transmitted. The receiver adapts the equalizer so that its output closely matches the known reference training signal. The more recent emergence of digital multipoint and broadcast systems has produced communication scenarios where training is infeasible or prohibited, since the inclusion of such signals sacrifices valuable channel capacity.*[5]*

Blind adaptive equalizers are those that do not need training to achieve convergence from an acceptable equalizer setting to a desired one. Blind equalization is desirable in multipoint and broadcast systems and necessary in noninvasive test and intercepts scenarios. Even in point-to-point communication systems, blind equalization has been adopted for various reasons, including capacity gain and procedural convenience.

There are basically two different approaches to the problem of blind equalization. The stochastic gradient descent (SGD) approach which iteratively minimizes a chosen cost function over all possible choices of equalizer coefficients, while the statistical approach uses sufficient stationary statistics collected over a block of received data for a channel identification or equalization. The latter approach often exploits higher order cyclo-stationary statistical information directly. The intended work is focused on blind equalization method using stochastic gradient approach.

Linear Blind Equalization System

The Least Mean Square (LMS) adaptive equalizer employing a training sequence is given by:

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu e(k)\mathbf{x}(k) \quad \text{.....1.1}$$

where μ is a small step size controlling the convergence of the algorithm, and $e(k)$ the difference between the output of the equalizer and the transmitted symbol. Naturally this algorithm requires that the channel input $a(k - v)$ be available, the equalizer iteratively minimizes the $E = |e(k)|^2$ mean square error(MSE) cost function in which the error is defined as :

$$e(k) = y(k) - a(k - v) \quad \text{.....1.2}$$

If the MSE is small such that after training the equalizer output $y(k)$ is a close estimate of the true channel input , then the decision device output can replace $a(k - v)$ in a decision directed algorithm that continues to track the modest time variations in the channel dynamics. In blind equalization the channel input $a(k)$ is unavailable, and thus different minimization criteria are explored. The crudest blind equalization scheme is the decision-directed scheme that updates the adaptive equalizer coefficients according to:

$$\mathbf{w(k+1) = w(k) + \mu(y_k - Q[y(k)])x(k)} \quad \text{.....1.3}$$

where $Q[y(k)] = \hat{a}(k - v)$ The ability of the equalizer to achieve desired convergence results when it is initialized with sufficiently small inter symbol interference (ISI) *[5]* accounts for the key role that decision-directed algorithm plays in channel equalization. Without direct training, a blind equalization algorithm is therefore used to provide a good initialization scheme for the decision-directed equalizer because of the decision-directed equalizer's poor convergence behavior under high ISI.

Thus a better adaptive algorithm is needed for the blind equalization of linear channels when the initial coefficients are far from ideal. The general structure of the blind adaptive algorithm is shown in the Figure 1.3. Blind Adaptive equalization algorithms are often designed by minimizing special non-MSE cost functions that do not directly involve the input $a(k)$ while still reflect the current level of ISI in the equalizer output.

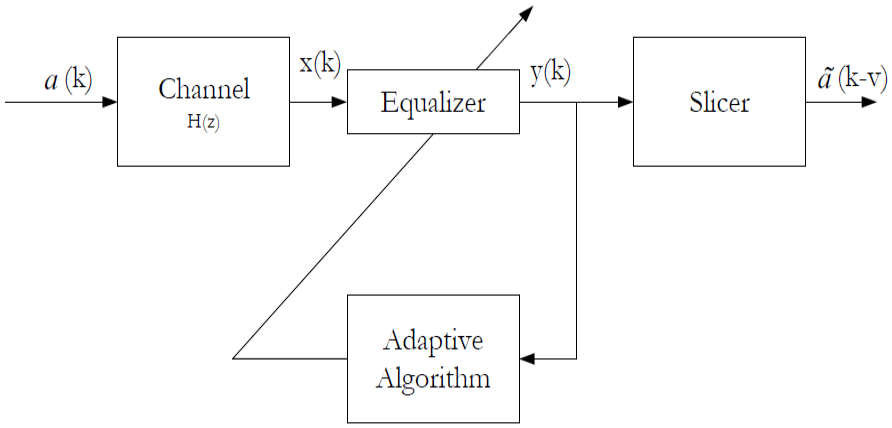


Fig 1.3: Linear Blind Equalization Systems

Let the mean cost function be defined as:

$$\mathbf{J_w = E\{\Psi(y(k))\}}$$

where Ψ (.) is a scalar function of the equalizer output. The mean cost function $J(w)$ should be specified such that its minimum, the corresponding w results in a minimum ISI or MSE equalizer. Because of the symmetric distribution of $\{a(k)\}$ over alphabet $\hat{A}$ the blind equalizer is unable to distinguish between $\pm a(k - v)$. Thus the function Ψ (.) should be even. In other words, both $y(k) = a(k- v)$ and $y(k) = -a(k - v)$ are acceptable objectives as global minima of the mean cost function. Using equation above, the stochastic gradient descent minimization algorithm is easily derived and is given by :

$$\mathbf{w(k+1) = w(k) + \mu \frac{\partial \Psi(y(k))}{\partial w}}$$

$$\mathbf{= w(k) + \mu \acute{\Psi}(w^H x(k) \ldots\ldots\ldots\ldots\ldots\ldots 1.4}$$

Where $\Psi' = \frac{\partial \Psi}{\partial x}$

The resulting blind equalization algorithm can be written as:

$w(k+1) = w(k) + \mu \Psi'(w^H(k)x(k))$ Hence, a blind equalizer can be defined by its cost function or its derivative. The derivative of the cost function is also called as the error function as it replaces the prediction error of the LMS algorithm.The Blind equalization when compared with the existing similar schemes gives better convergence rates at the cost of complexity.

CHAPTER 2 – ADAPTIVE FILTERS

2.1 Introduction

In this chapter, we make a comparison of the adaptive filters with other filters and discuss
the comparative advantages. We also study the adaptive filter theory*[8,9,10]* in detail, their types and applications. The chapter also includes the factors that determine the choice of an algorithm.

2.2 Types of Filters

2.2.1 Linear Optimum Filter

We may classify filters as linear or non-linear. A filter is said to be linear if the filtered, smoothed or, predicted quantity at the output of the filter is a linear function of the observations applied to the filter input. Otherwise, the filter is non-linear.

In the statistical approach to the solution of the linear filtering problem, we assume the availability of certain statistical parameters (i.e., mean and correlation functions) of the useful signal and unwanted additive noise, and the requirement is to design a linear filter with the noisy data as input so as to minimize the effects of noise at the filter output according to some statistical criterion. A useful approach to this filter-optimization problem is to minimize the mean-square value of the error signal, defined as the difference between some desired response and the actual filter output. For stationary inputs, the resulting solution is commonly known as the Wiener filter, which is said to be optimum in the mean-square error sense. The Wiener filter*[10]* is

inadequate for dealing with situations in which non stationery of the signal and /or noise is intrinsic to the problem. In such situations, the optimum filter has to assume a time varying form. A highly successful solution to this more difficult problem is found in the Kalman filter, which is a powerful system with a wide variety of engineering applications.

Linear filter theory, encompassing both Wiener and Kalman filters, is well developed in the literature for continuous–time as well as discrete-time signals*[11]*.

2.2.2 Adaptive Filters

As seen in last section, Wiener and Kalman filters are the mostly used filters. But, both of them have constraints, i.e., they require some priori information. Wiener filter requires knowledge of signal covariance, and Kalman filter requires knowledge of state-space model governing signal behavior.

In practice, such a priori information is rarely available; what is available is the data (sequence of numbers). Moreover, all the data is not available at a time; the data is coming in sequentially. This is where adaptive processing comes into play. The basic idea is to process the data as it comes in (i.e., recursively), and by a filter which is only data dependent, i.e., the filter parameters adapt to the coming data. Such filters are referred to as adaptive filters.

By such a system we mean one that is self-designing in that the adaptive algorithm, which makes it possible for the filter to perform satisfactorily in an environment where complete knowledge of the relevant signal characteristics is not available. The algorithm starts from some predetermined set of initial conditions, representing whatever we know about the environment. Yet, in a stationary environment, we find that after successive iterations of the algorithm it converges to the optimum Wiener solution in some statistical sense. In a non stationary environment, the algorithm offers a tracking capability, in that it can track time

variations in the statistics of the input data, provided that the variations are sufficiently slow.

As a direct consequence of the application of a recursive algorithm whereby the parameters of an adaptive filer are updated from one iteration to the next, the parameters become data dependent. This, therefore, means that an adaptive filter is in reality a non linear system, in the sense that it does not obey the principle of superposition. Notwithstanding this properly, adaptive filters are commonly classified as linear or non linear. An adaptive filter is said to be linear if its input-output map obeys the principle of superposition whenever its parameters are held fixed. Otherwise, the adaptive filter is said to be non linear.

2.3 Types of Adaptive Filters

The operation of a linear adaptive filtering algorithm involves two basic processes; (1) a filtering process designed to produce an output in response to a sequence of input data and (2) an adaptive process, the purpose of which is to provide a mechanism for the adaptive control of an adjustable set of parameters used in the filtering process. These two processes work interactively with each other. Naturally, the choice of a structure for the filtering process has a profound effect on the operation of the algorithm as a whole.

The impulse response of a linear filter determines the filter's memory. On this basis, we may classify filters into finite-duration impulse response (FIR)*[8]*, and infinite-duration impulse response (IIR) filters*[8]*, which are respectively characterized by finite memory and infinitely long, but fading, memory.

Although both IIR and FIR filters have been considered for adaptive filtering, the FIR filter is by far most practical and widely used. The reason for this preference is quite simple; the FIR filter has only adjustable zeros; hence, it is free of stability problems associated with adaptive IIR filter, which have adjustable poles as well as zeros. However, the stability of FIR filter depends critically on the algorithm for adjusting its coefficients.

2.4 Factors determining the choice of Algorithm

An important consideration in the use of an adaptive filter is the criterion for optimizing the adjustable filter parameters. The criterion must not only provide a meaningful measure of filter performance, but it must also result in a practically realizable algorithm.

A wide variety of recursive algorithms have been developed in the literature for the operation of linear adaptive filters. In the final analysis, the choice of one algorithm over another is determined by one or more of the following factors:

- *Rate of convergence.* This is defined as the number of iterations required for the algorithm, in response to stationary inputs, to converge "close enough" to the optimum Wiener solution in the mean-square error sense. A fast rate of convergence allows the algorithm to adapt rapidly to a stationary environment of unknown statistics.

- *Misadjustment.* For an algorithm of interest, this parameter provides a quantitative measure of the amount by which the final value of the mean-square error, averaged over an ensemble of adaptive filters, deviates from the minimum mean-square error produced by the Wiener filter.

- *Tracking.* When an adaptive filtering algorithm operates in a non stationary environment, the algorithm is required to track statistical variations in the environment. The tracking performance of the algorithm, however, is influenced by two contradictory features: (a) rate of convergence, and (b) steady-state fluctuation due to algorithm noise.

- *Robustness.* For an adaptive filter to be robust, small disturbances can only result in small estimation errors. The disturbances may arise from a variety of factors, internal or external to the filter.

Computational requirements. Here the issues of concern include (a) the number of operations required to make one complete iteration of the algorithm (b) the size of memory locations required to store the data and program, and (c) the investment required to program the algorithm on a computer.

Structure. This refers to the structure of information flow in the algorithm, determining the manner in which it is implemented in hardware form.

Numerical Properties. Numerical stability is an inherent characteristic of an adaptive filtering algorithm. Numerical accuracy, on the other hand, is determined by the number of bits used in the numerical representation of data samples and filter coefficients. An adaptive filtering algorithm is said to be numerically robust when it is insensitive to variations in the word length used in its digital implementation.

2.5 How to choose an Adaptive Filter

Given the wide variety of adaptive filters available to a system designer, the question arises how a choice can be made for an application of interest. Clearly, whatever the choice, it has to be cost effective. With this goal in mind, we may identify three important issued that require attention: computational cost, performance, and robustness.

Practical applications of adaptive filtering are highly diverse, with each application having peculiarities of its own. Thus, the solution for one application may not be suitable for another. Nevertheless, be successful, we have to develop a physical understanding of the environment in which the filter has to operate and thereby relate to the realities of the application of interest.

2.6 Applications of Adaptive Filter

The ability of an adaptive filter to operate satisfactorily in an unknown environment and track time variations of input statistics makes the adaptive filter a powerful device for signal processing and control applications. Indeed, adaptive filters have been successfully applied in such diverse fields as communications, radar, sonar, seismology, and biomedical engineering. Although these applications are quite different in nature, nevertheless, they have one basic feature in common: An input vector and a desired response are used to compute an estimation error, which is in turn used to control the values of a set of adjustable filter coefficients. The adjustable coefficients may take the form of tap weights, reflection coefficients, or rotation parameters, depending on the filter structure employed. However, the essential differences between the various applications of adaptive filtering arise in the manner in which the desired response is extracted. In this context, we may distinguish four basic classes of adaptive filtering
applications, as follows:

I. **Identification:**

I. a. *System Identification.* Given an unknown dynamical system, the purpose of system identification is to design an adaptive filter that provides an approximation to the system.

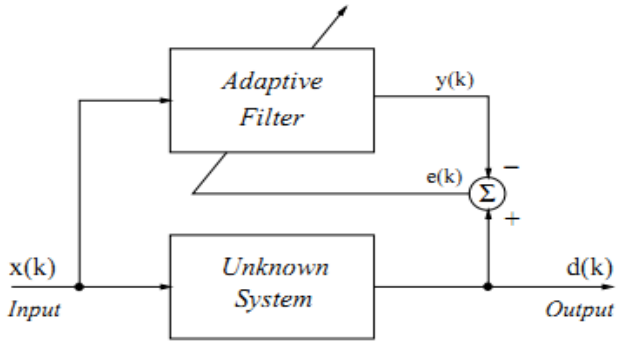


Fig: 2.1 Block diagram for system identification

I. b. *Layered Earth Modeling.* In exploration seismology, a layered model of the earth

is developed to unravel the complexities of the earth's surface.

II. Modeling:

II. a. *Equalization.* Given a channel of unknown impulse response, the purpose of an adaptive equalizer is to operate on the channel output such that the cascade connection of the channel and the equalizer provides an approximation to an ideal transmission medium.

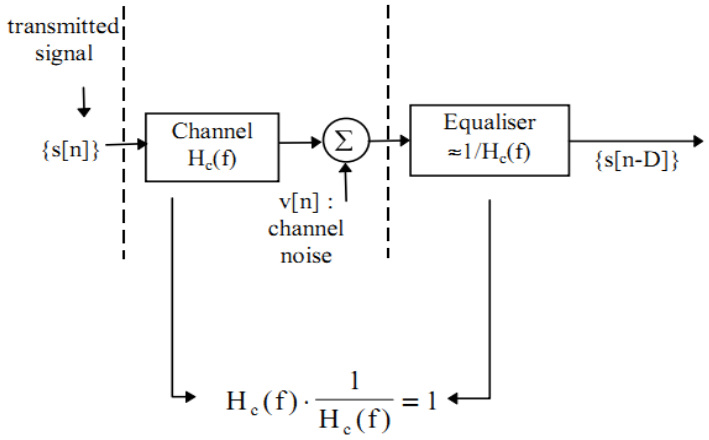


Fig : 2.2 Block diagram for a channel estimater.

III**. Prediction:**

III. a*. Predictive coding.* The adaptive prediction is used to develop a model of a signal of interest; rather than encode the signal directly, in predictive coding the prediction error is encoded for transmission or storage. Typically, the prediction error has a smaller variance than the original signal, hence the basis for improved encoding.

III. b*. Spectrum analysis.* In this application, predictive modeling is used to estimate the power spectrum of a signal of interest.

IV. **Interference cancellation:**

IV. a. *Noise cancellation.* The purpose of an adaptive noise canceller is to subtract noise from a received signal in an adaptively controlled manner so as to improve the signal-to-noise ratio.Echo cancellation, experienced on telephone circuits, is a special form of noise cancellation. Noise cancellation is also used in electrocardiography.

IV .b.*Beamforming.* A beamformer is a spatial filter that consists of an array of antenna elements with adjustable weights (coefficients). The twin purposes of an adaptive beamformer are to adaptively control the weights so as to cancel interfering signals impinging on the array from unknown directions and, at the same time, provide protection to a target signal of interest.

The application of adaptive filter considered in this project is Equalization, belonging to the Inverse modeling class of adaptive filtering application. Consider fig. 2.1, which illustrates the inverse modeling class of adaptive filtering application. The followingnotation is used in the figure:

x_k = input applied to adaptive filter;

y_k = output of the adaptive filter;

d_k = desired response; and

$e_k = d_k - y_k$ = estimation error.

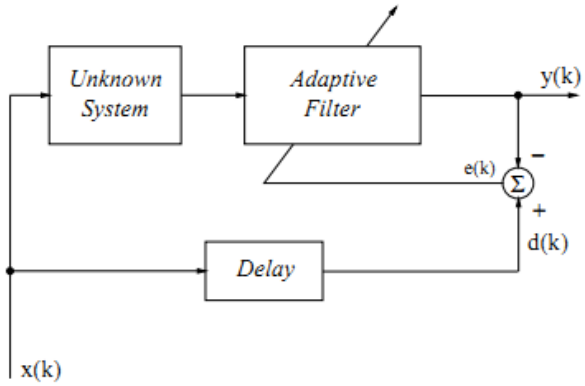


***Fig 2.3**: Inverse Modeling Class of Adaptive Filtering Applications.*

In inverse modeling, the function of the adaptive filter is to provide an inverse model that represents the best fit to an unknown noisy plant. Ideally in the case of an linear system, the inverse model has a transfer function equal to the reciprocal (inverse) of the plant's transfer function, such that the combination of the two constitutes an ideal transmission medium. A delayed version of the plant (system) input constitutes the desired response for the adaptive filter. In some applications, the plant input is used without delay as the desired response.

2.7 The Kalman Filter

The Kalman filter is an optimal linear minimum variance estimator. It can provide real-time estimates of the states of a system from noisy measurements. The Equations (2.2) and (2.1)describe a linear system and form a Kalman filtering problem. The algorithm given in Table 1 is well known for the Kalman Filter *[12,13,14]*. The estimate of $\boldsymbol{x}_k$ is $\hat{x}_k$ and $\boldsymbol{P}k$ is the error covariance matrix of state estimates.

The Kalman filter is a recursive algorithm composed of two parts: *Measurement Update Equations* and *Time Update Equations*. Using the measurement update equations, the Kalman filter estimates the next state vector of the linear system or the CIR based on a noisy measurement which is the input signal at the receiver. Then, using the time update equations, the Kalman filter updates its estimate of the

next state vector according to its knowledge of the linear system parameters such as ***F*** and ***G***.

The flow of instruction in the kalman filter is shown below in the figure 2.4 and in the table calculation of various filter parameters are shown in the form of equation. The true state evolves over time according to a true state-space model given Equation

$$\dot{\mathbf{x}}(\mathbf{t}) = \mathbf{F}(\mathbf{t})\mathbf{x}(\mathbf{t}) + \mathbf{B}(\mathbf{t})\mathbf{u}(\mathbf{t}) + \mathbf{G}(\mathbf{t})\mathbf{v}(\mathbf{t}) \ldots\ldots\ldots\ldots\ldots \mathbf{2.1}$$

with known input **u**(*k*) and subject to disturbances noise **v**(*t*). Observations are made of the true state according to a true observation model given by Equation.

$$\mathbf{z}(\boldsymbol{t}) = \mathbf{H}(\boldsymbol{t})\mathbf{x}(\boldsymbol{t}) + \mathbf{D}(\boldsymbol{t})\mathbf{w}(\boldsymbol{t}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{2.2}$$

The observations are sampled at discrete points in time and are subject to observation noises **w**(*k*). The cycle associated with the evolution of the true state shown in Figure 2.4 makes it clear that the true state is never known; the only output from this stage are the sampled observations made of the true state.

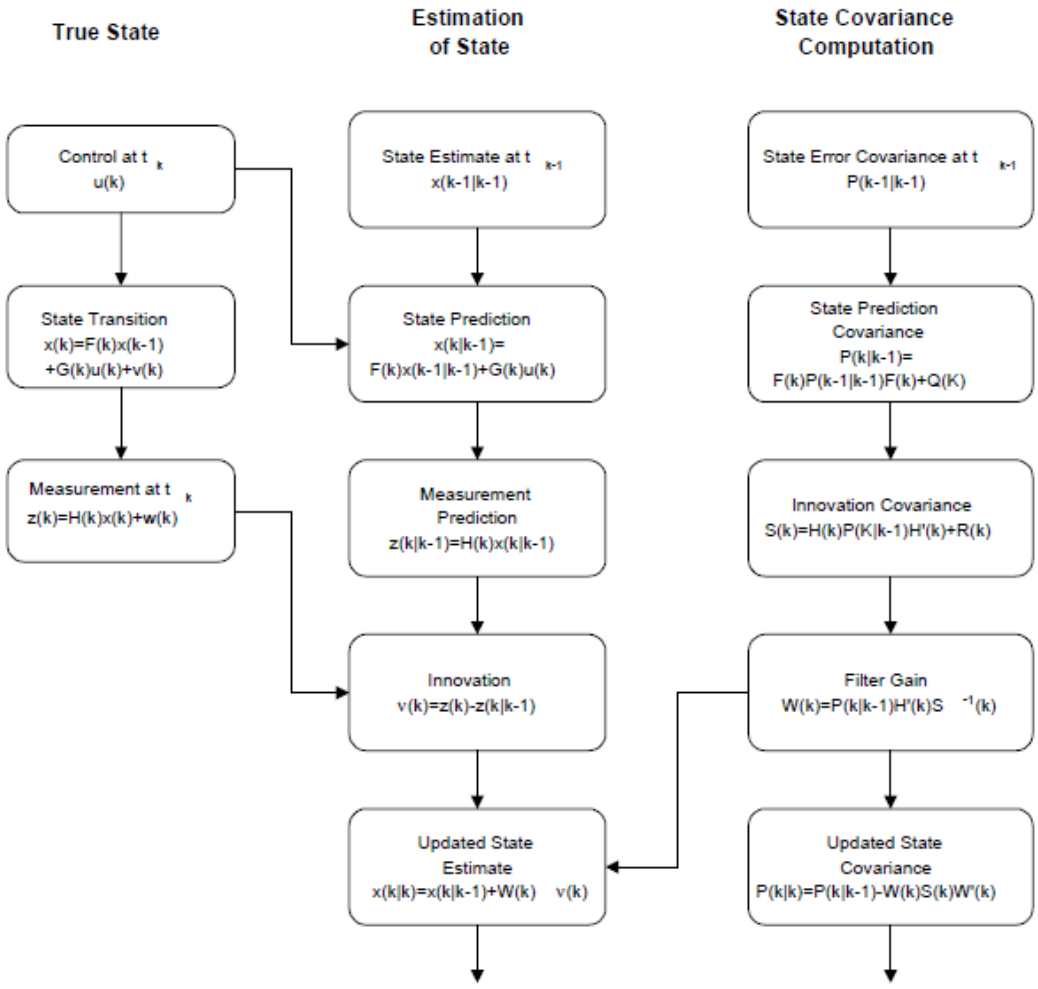


Fig : 2.4 flow of instruction in the kalman filtering process

The starting point for the estimation cycle is the generation of a state prediction from knowledge of the state estimate at the preceding time-step. This prediction is computed from Equation:

$$\hat{\mathbf{x}}(k \mid k-1) = \mathbf{F}(k)\hat{\mathbf{x}}(k-1 \mid k-1) + \mathbf{B}(k)\mathbf{u}(k) \ldots\ldots\ldots\ldots \mathbf{2.3}$$

and is based upon the known control input $\mathbf{u}(k)$ and the state transition model $\mathbf{F}(k)$. The state prediction is then used to compute a predicted observation from Equation:

$$\hat{\mathbf{z}}(\boldsymbol{k} \mid \boldsymbol{k} - \mathbf{1}) = \mathbf{H}(\boldsymbol{k})\hat{\mathbf{x}}(\boldsymbol{k} \mid \boldsymbol{k} - \mathbf{1}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{2.4}$$

according to the model **H**(*k*). This predicted observation is then tracted from the true observation according to Equation:

$$\boldsymbol{\nu}(\boldsymbol{k}) = \mathbf{z}(\boldsymbol{k}) - \mathbf{H}(\boldsymbol{k})\hat{\mathbf{x}}(\boldsymbol{k} \mid \boldsymbol{k} - \mathbf{1}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{2.5}$$

to give the innovation. The innovation is multiplied by the gain matrix (generated by the covariance loop) and added to the prediction to generate a state estimate according to Equation:

$$\hat{\mathbf{x}}(\boldsymbol{k} \mid \boldsymbol{k}) = \hat{\mathbf{x}}(\boldsymbol{k} \mid \boldsymbol{k} - \mathbf{1}) + \mathbf{W}(\boldsymbol{k})\boldsymbol{\nu}(\boldsymbol{k}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{2.6}$$

The time index is then incremented and the cycle repeated. It is important to note that the only inputs to this cycle are the control input **u**(*k*), the observation **z**(*k*) and the gain matrix **W**(*k*). The state and observation models **F**(*k*) and **H**(*k*) must also be specified. The primary output is the state estimate ˆ**x**(*k* | *k*).

The estimate covariance cycle also begins by generating a prediction covariance according to Equation:

$$\mathbf{P}(\boldsymbol{k} \mid \boldsymbol{k} - \mathbf{1}) = \mathbf{F}(\boldsymbol{k})\mathbf{P}(\boldsymbol{k} - \mathbf{1} \mid \boldsymbol{k} - \mathbf{1})\mathbf{F}^{\mathbf{T}}(\boldsymbol{k}) + \mathbf{G}(\boldsymbol{k})\mathbf{Q}(\boldsymbol{k})\mathbf{G}^{\mathbf{T}}(\boldsymbol{k}) \ldots\ldots\ldots\ldots \mathbf{2.7}$$

on the basis of the state model **F**(*k*) and the estimated process noise covariance **Q**(*k*). The innovation covariance is then computed according to Equation:

$$\mathbf{R}(\boldsymbol{k}) + \mathbf{H}(\boldsymbol{k})\mathbf{P}(\boldsymbol{k} \mid \boldsymbol{k} - \mathbf{1})\mathbf{H}^{\mathbf{T}}(\boldsymbol{k}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{2.8}$$

On the basis of the observation model $\mathbf{H}(k)$ and the estimated observation noise $\mathbf{R}(k)$. The innovation covariance, together with the prediction covariance, is then used to compute the gain matrix $\mathbf{W}(k)$ according to Equation

$$\boldsymbol{W}(\boldsymbol{k}) = \boldsymbol{P}(\boldsymbol{k} \mid \boldsymbol{k} - \mathbf{1}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{2.9}$$

The gain matrix is passed to the state estimation loop and is also used to compute the updated state covariance $\mathbf{P}(k \mid k)$ according to Equation 160. Finally the time index is incremented and the cycle is repeated. The only inputs to this covariance loop are the estimated process noise covariance $\mathbf{Q}(k)$ and the estimated observation noise covariance $\mathbf{R}(k)$. As in the estimation loop, the process model $\mathbf{F}(k)$ and the observation model must also be specified. The primary output from this loop is the estimate covariance $\mathbf{P}(k \mid k)$. It is important to note that the covariance loop is independent of the observations that are made and so is also independent of the evolution of the true state. This is simply because the information required by the covariance loop is the covariance of the process and observation noises and these are only available in terms of the estimated covariances $\mathbf{Q}(k)$ and $\mathbf{R}(k)$ respectively. One consequence of this is that all state covariance information can be computed off-line prior to any observations being made. It follows that all the gain matrices required by the state estimation cycle can also be computed off-line. This is significant because the majority of the computational requirements of the estimation process are dedicated to this task. As will be shown in a subsequent section, the state covariance tends to a steady-state value over time as, consequently, does the gain matrix. This fact can be used to construct estimators in which the gain matrix is time-invariant (and usually equal to its steady-state value) eliminating the need for the covariance loop and leading to significantly reduced computational requirements.

3.1 Equalization Theory

For a typical optical link, the received pulse shape, $h_r(t)$ is usually determined by the fiber impulse response, $h_f(t)$, and the transmitted pulse shape, $h_t(t)$.

$h_r(t) = h_t(t) * h_f(t)$……..3.1

where * denotes convolution. Normally, the transmitted pulse shape is known, leaving the engineer to estimate the impulse response of the fiber, which is often difficult to characterize. However, studies have shown that for a fiber that exhibits mode coupling, its impulse response is close to a cosine-squared shape in both the time and frequency domain.

$h_f(t) = e\ xp[-(t^2/(2(\alpha T)^2))] / [sqrt(2\pi)](\alpha T)$…….. 3.2

As such, it is highly probable that when a series of pulses is transmitted, overlapping will occur due to pulse broadening caused by dispersion as discussed in the previous section. Therefore, to reduce the pulse broadening that causes the resulting ISI, a suitable equalizer with a frequency response of $H_{eq}(w)$ may be implemented.

$H_{eq}(\omega) = H_{out}(\omega) / H_A(\omega)$…….. 3.3

where $H_{out}(\omega) = F[h_{out}(t)]$, is the desired output pulse shape, and $H_A(\omega) = F[h_A(t)]$, is the total dispersive response of the system and F denotes Fourier transform.

As equalization makes up one of the main parts of an optical receiver together with the detector and amplifier, the equalizer is often designed to include the effects of the channel as well as the degradations caused by the amplifier.

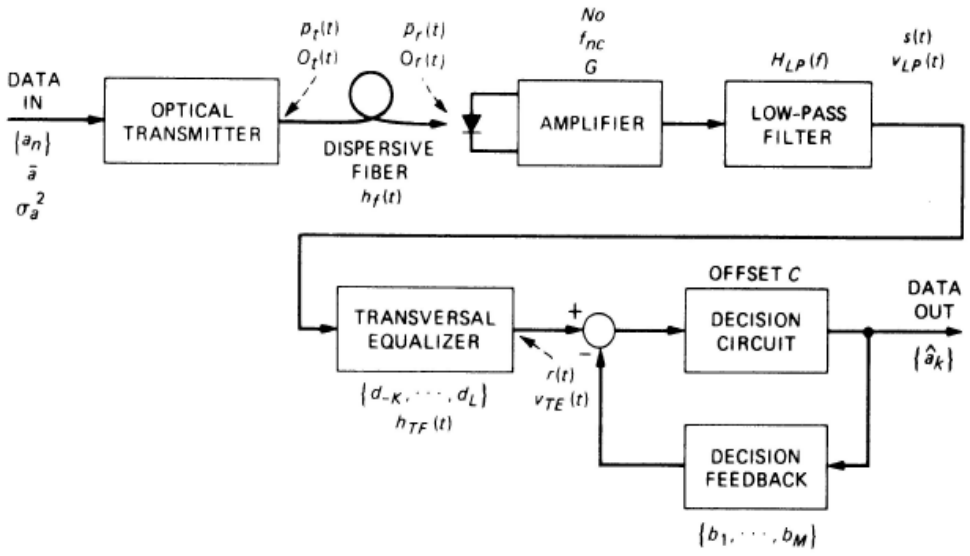


Figure 3.1 Application of a Decision Feedback Equalizer in an optical transmission system

3.2 Adaptive Filter Theory

Adaptive equalization is increasing popular in optical fiber communications. However, with the ever-increasing demand for larger bandwidth and faster transmission speed, the increase in distortion in an optical channel is likely to be significant. Therefore, it is gainful to delve into adaptive equalization techniques to suppress distortion in an optical communication channel. Minimizing distortion will in turn, allow for longer haul communication before requiring a repeater, which will save infrastructure and equipment cost for a communication link. Adaptive filters are systems that can adjust

themselves to different environments. It involves a process of filtering some input signal to match a desired response.

The filter parameters are updated by making a set of measurements of the underlying signals and applying that set to the adaptive filtering algorithm such that the difference between the filter output and the desired response is minimized in either a statistical or deterministic sense.

A simple yet effective illustration of the principle behind adaptive filtering is given in Fig (3.2).

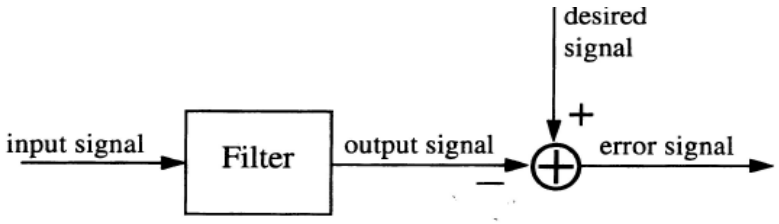


Figure 3.2: A simple diagram on the principle of adaptive filtering.

The general operation of an adaptive equalizer in an optical communication channel is to track the variations of the laser and fiber response over time. This is achieved by sending a known training sequence through the channel and obtaining the difference when the output is subtracted from the known sequence. The difference is known as the prediction error. The computed error is then used to adjust the tap coefficients so that the channel response can be estimated. This is done recursively using an adaptive algorithm like the least mean square (LMS) algorithm. Once the equalizer has converged, the actual data can then be sent and received accurately as the channel response is known and compensated for by the equalizer.

3.3 Comparison of Adaptive Algorithms

There exist a great number of adaptive algorithms, each with its own unique properties and applications. Before looking into the types of adaptive algorithms available, it is useful to look into factors that determine the performance of such algorithms.

Convergence Rate – This is defined as the number of iterations required for the algorithm to converge close enough to the optimum solution. A fast convergence rate enables the algorithm to track statistical variations when operating in a non stationary environment and requires a shorter training sequence.

Computational Complexity – This is defined as the number of operations required for the algorithm to make one complete iteration.

Misadjustment – this parameter provides a quantitative measure of the amount by which the final value of the mean square error, averaged over an ensemble of adaptive filters, deviates from the optimal minimum mean square error.

Numerical properties – when an algorithm is implemented numerically, inaccuracies are produced due to round-off noise and representation errors in the computer. This influences the stability of the algorithm.

3.4 Types of Equalizers suitable for an Optical Channel

In digital communications, an equalizer is a device that attempts to recover a signal transmitted through an ISI channel. It may be a simple linear filter or a complex algorithm.

Several equalizer types are listed below:

- Linear Equalizer: processes the incoming signal with a linear filter
- MMSE equalizer: designs the filter to minimize $E[|e|^2]$, where e is the error signal, which is the filter output minus the transmitted signal.
- Zero Forcing Equalizer: approximates the inverse of the channel with a linear

filter.

- Decision Feedback Equalizer: augments a linear equalizer by adding a filtered version of previous symbol estimates to the original filter output.

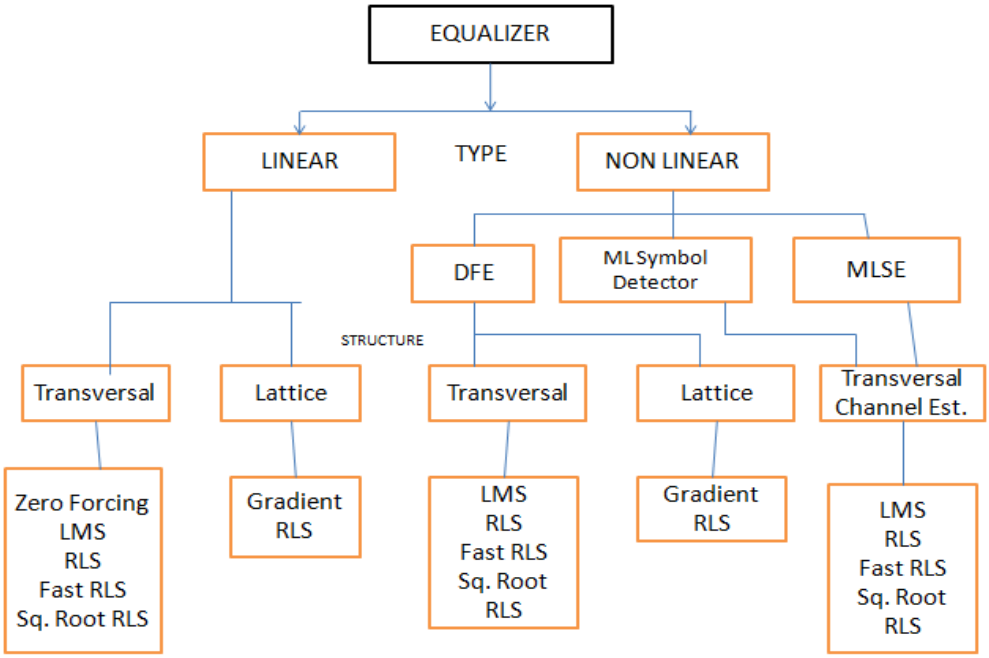


Fig 3.3: Classification of Equalizers

- Blind Equalizer: estimates the transmitted signal without knowledge of the channel statistics, using only knowledge of the transmitted signal's statistics.
- Adaptive Equalizer: is typically a linear equalizer or a DFE. It updates the equalizer parameters (such as the filter coefficients) as it is processes the data. Typically, it uses the MSE cost function; it assumes that it makes the correct symbol decisions, and uses its estimate of the symbols to compute , which is defined above.
- Viterbi Equalizer: Finds the maximum likelihood (ML) optimal solution to the equalization problem. Its goal is to minimize the probability of making an error over the entire sequence

The inherent property behind linear equalizers is that they are optimum with respect to the criterion of minimum probability of symbol error when the channel does not suffer from amplitude distortion. However, in a practical optical communication channel, amplitude distortion is one of the major detrimental effects.

Therefore, the investigation and research into the Decision Feedback Equalizer, which is a non-linear equalizer and capable of superior performance in amplitude distorted channels, would be very beneficial and relevant to the application in optical communications. When implementing the DFE structure, enhancements like the Fractionally Spaced Equalization, which makes the equalizer more robust to amplitude distortions, can also be considered.*[15]*

3.5 Working principle of adaptive filters

The adaptive filter operates on the input x[n] to produce an estimate of the desired response d[n]. The generation of the desired response is an important issue. To measure the performance of an adaptive filter we can consider how functions of the error J(e[n]) behave as time increases, or whether the filter coefficient (weight) vector w(n) approaches some optimal setting.

Adaptive Algorithm

All algorithms are based on minimising some function of the error : -

$J(e[n]) = e^2$.

$e[n] = d[n] - y[n] = d[n] - x^T[n]\, w[n] = d[n] - w^T[n]\, x[n]$

The error squared form will be found to be most analytically tractable and appropriate for measurements corrupted by Gaussian noise. When the measurement noise is sub -Gaussian higher power errors are preferred whilst for super-Gaussian measurement noise distributions, lower power errors are more useful. To derive the optimal setting of an adaptive FIR filter we shall assume that the input and derived response signals are zero-mean and WSS. The function we wish to minimise is the Mean Square Error (MSE) : -

$J = E\{e^2[n]\}$

Variables in error function are

w(n) = the p×1 column weight (parameter) vector of the filter

= [$w_1(n)$, $w_2(n)$, . . . , $w_p(n)$] ^{T}

x[n] = [x[n] , x[n − 1] , . . . , x[n − p + 1]] ^{T} the input vector

so

$$\boldsymbol{y[n] = X[n]^H w} \text{.....3.4}$$

Thus, the cost (error , objective) function becomes: -

$$J(W) = E\{[d[n] - y[n]]^2\} \text{ ... ... 3.5}$$

$$J(W) = \sigma_d^2 - p^H W - W^H p + W^H R W \text{ ... ... 3.6}$$

Where autocorrelation $\boldsymbol{R = E\{X[n]X[n]^T\}}$ **3.7**

And cross correlation $\boldsymbol{p = E\{X[n]d[n]\}}$ **3.8**

Error performance surface:

Thus ***J(w)*** is quadratic function of ***w*** and provided that ***R*** is full rank, ***J(w)*** will have one unique minimum.

Consider a filter with two parameters such that w = [w(1), w(2)]^T .

The input {x[n]} is assumed to be zero mean , white noise , so that the contours of constant J, as in the figure on the right below , are circular.

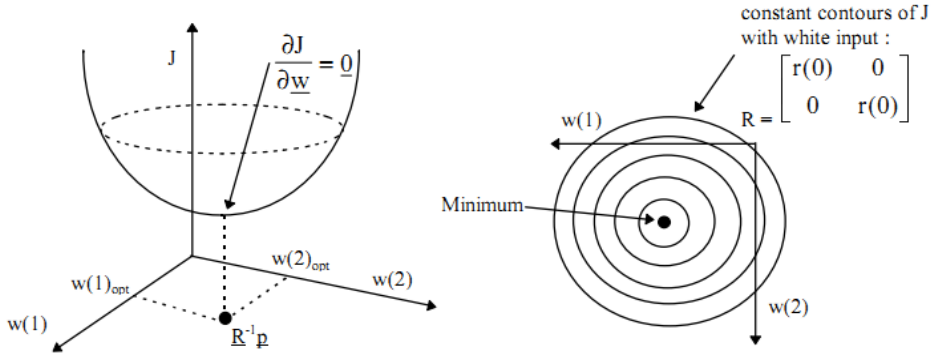


Fig: 3.4 contour of J

3.5.1Wiener Hopf Equation:

- In order to minimise the function ***J(w)*** . Using *Wiener Filtering* we write J(w) as perfect square in w

$$J(W) = \sigma_d^2 - p^H R^{-1} p + (W - R^{-1}p)^H R (W - R^{-1}p) \dots\dots\dots 3.9$$

- Then by substituting W as

  $W - R^{-1}p$.....3.10

  $J(W) = \sigma_d^2 - p^H R^{-1} p \dots\dots\dots$.3.11

Another Method:

- In case of *Complex variables derivatives are defined as*

$$\frac{\partial y}{\partial x} = \frac{1}{2}\left[\frac{\partial y}{\partial x_{\mathrm{Re}}} - j\frac{\partial y}{\partial x_{\mathrm{Im}}}\right]$$

$$for\ y = x$$

$$\frac{\partial y}{\partial x} = \frac{1}{2}\left[\frac{\partial\left(x_{\mathrm{Re}} + jx_{\mathrm{Im}}\right)}{\partial x_{\mathrm{Re}}} - j\frac{\partial\left(x_{\mathrm{Re}} + jx_{\mathrm{Im}}\right)}{\partial x_{\mathrm{Im}}}\right] = 1$$

$$for\ y = x^*$$

$$\frac{\partial y}{\partial x} = \frac{1}{2}\left[\frac{\partial\left(x_{\mathrm{Re}} - jx_{\mathrm{Im}}\right)}{\partial x_{\mathrm{Re}}} - j\frac{\partial\left(x_{\mathrm{Re}} - jx_{\mathrm{Im}}\right)}{\partial x_{\mathrm{Im}}}\right] = 0$$

- Take *derivative of J(w) with respect to w and* w^H

$$\nabla J(\mathbf{w}) = \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} + \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}^H} = -2\mathbf{p} + 2\mathbf{R}\mathbf{w} = 0$$

$$\mathbf{w}_0 = \mathbf{R}^{-1}\mathbf{p}.....3.12$$

- *Wiener Hopf Equation*

$$\mathbf{w}_0 = \mathbf{R}^{-1}\mathbf{p}$$

$$\min_{\mathbf{w}} J(\mathbf{w}) = \sigma_d^2 - \mathbf{p}^H \mathbf{R}^{-1}\mathbf{p}.....3.13$$

Table 1 summary of wiener filter algorithm

Summary of Wiener filter Algorithm	
Initialization	Initialize Filter coefficients $\hat{w}(0)=0$ Initialize Input vector x(n)=0
Inputs	Filter coefficients vector estimate $\hat{w}(n)$ Input vector x(n) Desired output d(n) Autocorrelation Matrix $R(n)$
Output	Filter output $\hat{y}_{n-1}(n)$ Update filter coefficient vector $\hat{w}(n+1)$
Algorithm:	
1.Get Autocorrelation Matrix $R(n)=x(n)\times x^T(n)$	
2. Get Cross correlation Matrix $p(n)=x(n)\times d(n)$	
3. Output (filtering) $\hat{y}(n)=\hat{w}^T(n)x(n)$	
4. Coefficient vector Updating $\mathbf{w}[n+1]=\mathbf{R}^{-1}(n)\mathbf{p}(n)$	

In the above table the important steps for the wiener filter is given.

3.5.2 Role of Eigen-Analysis in Wiener Solution

The shape of the error performance surface is related directly to the eigen–structure of the autocorrelation matrix R. The condition number of R , that is, $\lambda_{max}/\lambda_{min}$, is particularly important.

For a white input,

$$R=\begin{bmatrix} r_{xx}(0) & 0 \\ 0 & r_{xx}(0) \end{bmatrix} \quad \rightarrow condition \quad number=1$$

Therefore the contours of J are circles when projected on to the [w(1), w(2)] plane.

When the input is coloured, the condition number increases, and the contours will take an elliptical form**.**

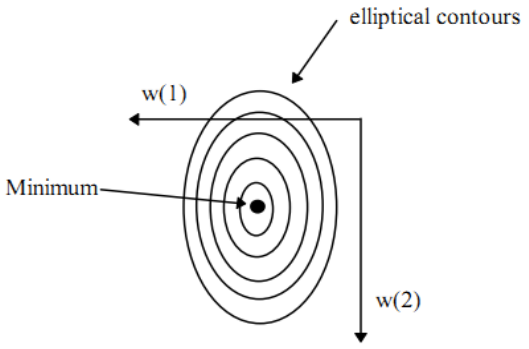


Fig: 3.5 contour of J according to eigen values

3.6 Steepest Descent algorithm

The update equation for the steepest descent method to find the minimum of some function J is given by

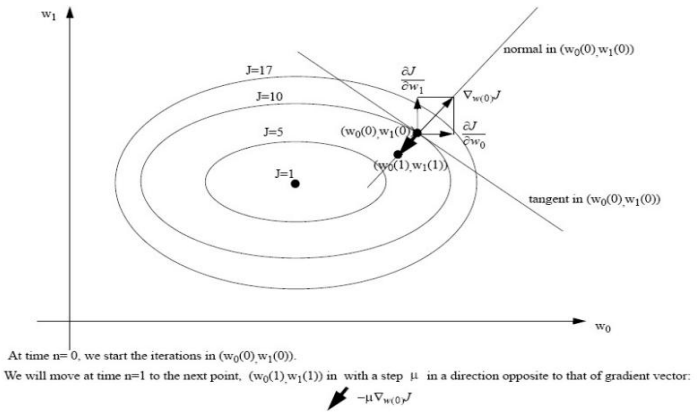
$$\mathbf{w}[n+1] = \mathbf{w}[n] - \frac{1}{2}\mu\frac{\partial J(\mathbf{w}[n])}{\partial \mathbf{w}[n]} \quad \text{.....3.14}$$

And $$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{p} + 2\mathbf{R}\mathbf{w} \quad \text{.....3.15}$$

- So that the *"weights update equation"*

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \mu(\mathbf{p} - \mathbf{R}\mathbf{w}[n]) \quad \text{.....3.16}$$

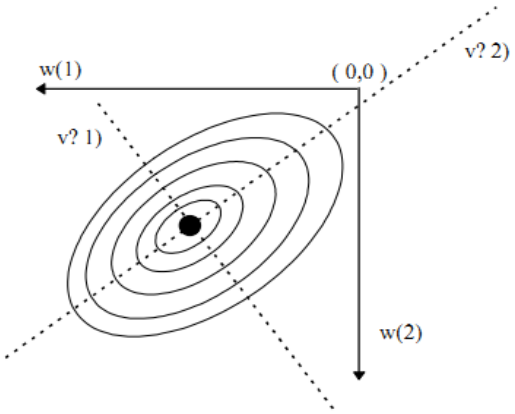
The parameter μ is termed the adaptation gain (learning rate, step size) and controls the speed of convergence



The convergence of the Steepest Descent algorithm from the initial point w[0] toward the optimum

Coloured Input–Convergence

For the coloured case, as depicted in the figure below, the direction of steepest descent does not necessarily point at the minimum, it depends on the starting point we are taking.To analyse the convergence of the method of steepest descent , replace the [w(1), w(2)] axis by moving the origin to w_{opt} and replacing w by $v=(w-w_{opt})$ and then rotating the axes by a new matrix S, to align with the principal axes denoted v ′ in the diagram above.



Eigen values and convergence

The matrix S corresponds to a component of the spectral factorisation of the autocorrelation matrix, i.e.

$R_{xx} = S\Lambda S^T$ where $\Lambda = \text{Diag}\ (\lambda_1,\lambda_2, \ldots, \lambda_p)$

and $S = [s_1,s_2, \ldots ,s_p]$, s_i is a normalised eigen vector.

Therefore $S \times S^T = I$ the $p \times p$ identity matrix.

The purpose of re defining the axes is to “decouple” the learning modes of the adaptive filter.

Proceeding with the analysis of w[n + 1], we have

$$w[n + 1] = w[n] + \mu(p - R\, w[n])$$

$= w[n] + \mu(R\, w_{opt} - Rw\,[n])$

$= w[n] + \mu\, R(w_{opt} - w[n])$…….....3.17

3.6.1Convergence analysis – weight error vector v(n)

- The error in the weights with respect to their optimal values is given by (using the Wiener solution for **p**)

$$\mathbf{w}[n+1] - \mathbf{w}_{opt} = \mathbf{w}[n] - \mathbf{w}_{opt} + \mu(\mathbf{R}\mathbf{w}_{opt} - \mathbf{R}\mathbf{w}[n]) \text{.....3.18}$$

We obtain $\mathbf{e}_h[n+1] = \mathbf{e}_h[n] - \mu\mathbf{R}\mathbf{e}_h[n]$.....3.19

Equivalently $\mathbf{e}_h[n+1] = (1 - \mu\mathbf{Q}^{\mathbf{T}}\mathbf{\Lambda}\mathbf{Q})\mathbf{e}_h[n]$.....3.20

i.e.
$$\mathbf{Q}\mathbf{e}_h[n+1] = \mathbf{Q}(1 - \mu\mathbf{Q}^{\mathbf{T}}\mathbf{\Lambda}\mathbf{Q})\mathbf{e}_h[n]$$
$$= (\mathbf{Q} - \mu\mathbf{Q}\mathbf{Q}^{\mathbf{T}}\mathbf{\Lambda}\mathbf{Q})\mathbf{e}_h[n]$$

thus we have $\mathbf{Q}\mathbf{e}_\mathbf{h}[\mathbf{n}+\mathbf{1}] = (\mathbf{1} - \mu\mathbf{\Lambda})\mathbf{Q}\mathbf{e}_\mathbf{h}[\mathbf{n}]$ … … … … . . **3. 21**

- Form a new variable $\mathbf{v}[n] = \mathbf{Q}\mathbf{e}_h[n]$

So that $\mathbf{v}[n+1] = (1 - \mu\mathbf{\Lambda})\mathbf{v}[n]$

Finally, v(n +1) = [I − μΛ]v(n) where I− μΛ is diagonal matrix

and we have the so-called modes of convergence

$v_j[\, n + 1] = (1 - \mu\lambda_j)v_j(n)$ where j = 1, 2, . . . , p

For each mode , at adaptation sample number n, we have : -

$v_j[\, n + 1] = (1 - \mu\lambda_j)^n v_j(0)$

For convergence , we require that

$| 1 - \mu\, \lambda_j | < 1$

then the algorithm is guaranteed to converge to the **Wiener –Hopf Solution**

$|1 - \mu\, \lambda_j| < 1 \Rightarrow -1 < 1 - \mu\, \lambda_j < 1$

Now : - $0 < \mu < 2/\lambda_i \ \forall \ \lambda_i$

Generally, the eigen values are not equal ($\lambda_j = \sigma^2$ N if white noise $\forall$ i) ,

therefore we take the worst case

$$0 < \mu < \frac{2}{\lambda_{\max}} \text{.....3.22}$$

This condition is also sufficient for convergence of the steepest descent algorithm in the mean square.

Table 2 Summarry of steepest descent method

Summary of Steepest Descent Algorithm	
Initialization	Initialize Filter coefficients $\hat{w}(0) = 0$ Initialize Input vector x(n)=0
Inputs	Filter coefficients vector estimate $\hat{w}(n)$ Input vector x(n) Desired output d(n) Autocorrelation Matrix $R(n)$
Output	Filter output $\hat{y}_{n-1}(n)$ Update filter coefficient vector $\hat{w}(n+1)$
Algorithm:	
1.Get Autocorrelation Matrix $R(n) = x(n) \times x^T(n)$ Get Maximum eigen value of Autocorrelation Matrix $\lambda_{\max}$	
2. Get Cross correlation Matrix $p(n) = x(n) \times d(n)$	
3.Calculate step size as $0 < \mu < \frac{2}{\lambda_{\max}}$	
4. Output (filtering) $\hat{y}(n) = \hat{w}^T(n)x(n)$	
5. Coefficient vector Updating $\mathbf{w}[n+1] = \mathbf{w}[n] + \mu(\mathbf{p} - \mathbf{R}\mathbf{w}[n])$	

3.7 The Least Mean Square (LMS) Algorithm

From the steepest descent algorithm

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{p} + 2\mathbf{R}\mathbf{w}$$

And $$\mathbf{w}[n+1] = \mathbf{w}[n] + \mu\left(-\nabla J\,|_{\mathbf{w}[n]}\right)$$

- So that the *"weights update equation"*

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \mu(\mathbf{p} - \mathbf{R}\mathbf{w}[n]).....3.23$$

The parameter μ is termed the adaptation gain (learning rate, step size) and controls the speed of convergence

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \mu(\mathbf{x}[n](d[n] - \mathbf{x}[n]^H\,\mathbf{w}[n]))$$

$$\mathbf{w}[n+1] = \mathbf{w}[n] + \mu e[n]\mathbf{x}[n].....3.24$$

Table 7.2.1 The LMS Algorithm for an M[th]-Order FIR Adaptive Filter

Inputs:	M = filter length μ = step-size factor $\mathbf{x}(n)$ = input data to the adaptive filter $\mathbf{w}(0)$ = initialization filter vector = $\mathbf{0}$
Outputs:	$y(n)$ = adaptive filter output = $\mathbf{w}^T(n)\mathbf{x}(n) \equiv \hat{d}(n)$ $e(n) = d(n) - y(n)$ = error $\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu e(n)\mathbf{x}(n)$

Table 3 LMS algorithm

Computational requirement for the LMS algorithm:-

• To calculate e[n]

p multiplications + *p* additions

• For weight update

- 1 multiplication (f or 2μ e[n]) + *p* multiplications (f or μx [n]e[n]) ⇒(*p* + 1) multiplications
- *p* additions (updating w[n])

⇒ the LMS algorithm is an O(2N) algorithm

• only twice the complexity of a fixed filter

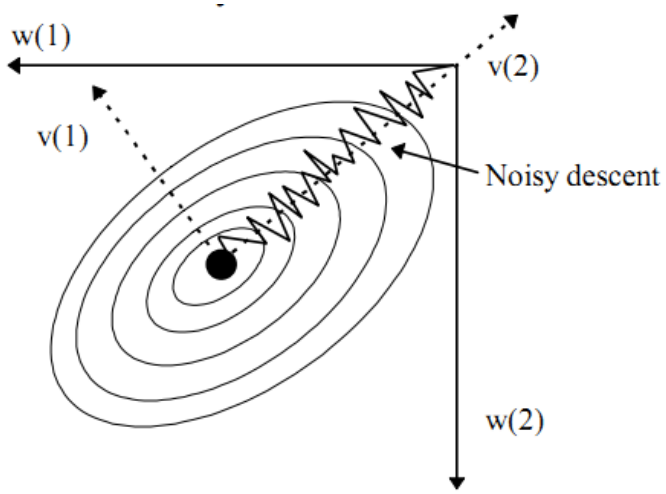
• together with its robust performance , is the reason why it finds extensive use in channel equalisation and echo cancellation in modems, and coding in speech (ADPCM) codecs.

3.7.1 Error performance surface for LMS

The actual LMS algorithm follows a noisy descent direction due to the approximate gradient expression used in the update equation.

Only on the average will the LMS algorithm follow the direction of Spectral Density.

We wish to determine the value so that the average value of w[n] tends to the Wiener solution- this does not mean that the actual value of w[n] will equal the Wiener solution at any time.



3.8 Recursive Least Squares Algorithm

To implement the recursive method of least squares, we start the computation with known initial conditions and then update the old estimate based on the information contained in data samples. Next, we minimize the cost function *J(n)*, where n is variable length of observed data

$J(n) = \sum_{k=1}^{n} \eta_n(k) e^2(k), \qquad \eta_n(k) \equiv \textit{weighting vector}$

where

$e(k) = d(k) - y(k) = d(k) - w^H(k)\mathbf{x}(k)$

$\mathbf{x}[k] = [x[k] \quad x[k-1] \ldots\ldots\ldots \quad . \quad x[k-m+1]]^T$ 3.25

$\mathrm{w} = [w[0] \quad w[1] \ldots\ldots\ldots \quad . \quad w[m+1]]^T$ 3.26

In standard RLS algorithm, the weighting factor $\eta_n(k)$ is choosen to have exponential form $\eta_n(k) = \lambda^{n-k}$ k=1,2,3….n

$$J(n) = \sum_{k=1}^{n} \lambda^{n-k} e^2(k) \ldots\ldots\ldots\ldots\ldots\ldots 3.27$$

Where the value of λ is less than one and, hence $\eta_n(k)$ is confirmed in range $0 < \eta_n(k) \leq 0$ for k=1,2,….n. the weighting factor $\eta_n(k)$ is also known as forgetting factor, since it weights (emphasizes) the recent data and tends to forget the past.

The minimum value of *J(n)* is attained when the normal equation

$R_\lambda(n)\hat{w} = p_\lambda(n)$ are satisfied and where the M×M correlation matrix $R_\lambda(n)$ is defined as

$$R_\lambda(n) = \sum_{k=1}^{n} \lambda^{n-k} \mathbf{x}(k)\mathbf{x}^T(k) = \mathrm{X^T \Lambda X}$$

$$P_\lambda(n) = \sum_{k=1}^{n} \lambda^{n-k} \mathbf{x}(k) d(k) = \mathrm{X^T \Lambda d}; \qquad \Lambda = diag[\lambda^{n-1} \quad \lambda^{n-2} \quad \ldots\ldots 1]$$

Note that R_λ differs from R in the following respect:

- The common matrix $x^T[n]\,x[n]$ is weighted by the exponential factor λ^{n-k}.

$$R_\lambda(n)=\lambda\left[\sum_{k=1}^{n-1}\lambda^{n-1-k}\mathbf{x}(k)\mathbf{x}^T(k)\right]+\mathbf{x}(n)\mathbf{x}^T(n)$$

$$R_\lambda(n)=\lambda R_\lambda(n-1)+\mathbf{x}(n)\mathbf{x}^T(n).....3.28$$

similarly

$$p_\lambda(n)=\lambda p_\lambda(n-1)+\mathbf{x}(n)d(n).....3.29$$

Matrix Inversion Lemma:-

$$A+xy^T=A\left(I+A^{-1}xy^T\right)$$

$$\left(A+xy^T\right)^{-1}=\left(I+A^{-1}xy^T\right)^{-1}A^{-1}.....3.30$$

$$Assume\,(I+A)^{-1}$$

$$(I+A)^{-1}=I-A+A^2-A^3+.....$$

$$\left(I+A^{-1}xy^T\right)^{-1}=I-A^{-1}xy^T+\left(A^{-1}xy^T\right)^3-\left(A^{-1}xy^T\right)^3+....$$

$$\left(I+A^{-1}xy^T\right)^{-1}=I-A^{-1}xy^T+A^{-1}xy^TA^{-1}xy^T-....$$

$$\left(A+xy^T\right)^{-1}=A^{-1}-A^{-1}xy^TA^{-1}+A^{-1}\left(y^TA^{-1}x\right)y^TA^{-1}-....$$

$$\left(A+xy^T\right)^{-1}=A^{-1}-A^{-1}xy^TA^{-1}\left(I-y^TA^{-1}x+\left(y^TA^{-1}x\right)^2-...\right)$$

$$\left(A+xy^T\right)^{-1}=A^{-1}-\frac{A^{-1}xy^TA^{-1}}{1+A^{-1}xy^TA^{-1}}.....3.31$$

Applying above lemma on following equation

$$\left(\lambda A+xx^T\right)^{-1}=\lambda^{-1}A^{-1}-\frac{\left(\lambda^{-1}A^{-1}\right)xx^T\left(\lambda^{-1}A^{-1}\right)}{1+\lambda^{-1}x^TA^{-1}x}.....3.33$$

Applying above lemma in calculating inverse of R_λ

$$R_\lambda^{-1}(n) = \lambda^{-1}R_\lambda^{-1}(n-1) + \frac{\lambda^{-2}R_\lambda^{-1}(n-1)\mathbf{x}(n)\mathbf{x}^T(n)R_\lambda^{-1}(n-1)}{1+\lambda^{-1}\mathbf{x}^T(n)R_\lambda^{-1}(n-1)\mathbf{x}(n)} \quad \text{.....3.34}$$

let $g(n) = \dfrac{\lambda^{-1}R_\lambda^{-1}(n-1)\mathbf{x}(n)}{1+\lambda^{-1}\mathbf{x}^T(n)R_\lambda^{-1}(n-1)\mathbf{x}(n)}$

$$= \frac{R_\lambda^{-1}(n-1)\mathbf{x}(n)}{\lambda+\mathbf{x}^T(n)R_\lambda^{-1}(n-1)\mathbf{x}(n)} \quad \text{.....3.35}$$

from $(1)\&(2)$ *we obtain*

$$R_\lambda^{-1}(n) = \lambda^{-1}\left[R_\lambda^{-1}(n-1) - g(n)\mathbf{x}^T(n)R_\lambda^{-1}(n-1)\right] \quad \text{.....3.36}$$

rearranging (2) *we get*

$$g(n) = \lambda^{-1}\left[R_\lambda^{-1}(n-1) - g(n)\mathbf{x}^T(n)R_\lambda^{-1}(n-1)\right]\mathbf{x}(n) \quad \text{.....3.37}$$

from $(3)\&(4)$ *we obtain*

$$g(n) = R_\lambda^{-1}(n)\mathbf{x}(n) \quad \text{.....3.38}$$

considering $R_\lambda(n)\hat{w}(n) = p_\lambda(n)$

and $p_\lambda(n) = \lambda p_\lambda(n-1) + \mathbf{x}(n)d(n)$

$$\hat{w}(n) = \lambda R_\lambda^{-1}(n)p_\lambda(n) + R_\lambda^{-1}(n)\mathbf{x}(n)d(n)$$

$$= \lambda R_\lambda^{-1}(n)p_\lambda(n)g(n)d(n) \quad \text{.....3.39}$$

from $(6)\&(3)$ *we obtain*

$$\hat{w}(n) = \hat{w}(n-1) + g(n)\left[d(n) - \hat{w}^T(n-1)\mathbf{x}(n)\right] \quad \text{.....3.40}$$

apriori estimation error is given by

$$\hat{e}_{n-1}(n) = d(n) - \hat{w}^T(n-1)\mathbf{x}(n) \quad \text{.....3.41}$$

$$\hat{w}(n) = \hat{w}(n-1) + g(n)\hat{e}_{n-1}(n) \quad \text{.....3.42}$$

apriori estimation error is given by

$$\hat{e}_n(n) = d(n) - \hat{w}^T(n)\mathbf{x}(n) \quad \text{.....3.43}$$

Summary of Recursive Least Square Algorithm	
Initialization	$R_\lambda(0) = \delta^{-1} I, \quad \delta < 0.01\sigma_x^2 = Variance\ of\ the\ data$ $\hat{w}(0) = 0$
Inputs	Filter coefficients vector estimate $\hat{w}(n-1)$ Input vector x(n) Desired output d(n) Matrix $R_\lambda^{-1}(n-1)$
Output	Filter output $\hat{y}_{n-1}(n)$ Update filter coefficient vector $\hat{w}(n)$ Update matrix $R_\lambda^{-1}(n)$
Algorithm:	
1.Gain vector computation $\bar{g}(n) = R_\lambda^{-1}(n-1)x(n),\ g(n) = \frac{1}{\lambda + x^T(n)\bar{g}(n)}\bar{g}(n)$	
2.Output (filtering) $\hat{y}_{n-1}(n) = \hat{w}^T(n-1)x(n)$	
3.Error estimate $\hat{e}_n(n) = d(n) - \bar{y}_{n-1}(n) = d(n) - \hat{w}^T(n)\mathbf{x}(n)$	
4. Coefficient vector Updating $\hat{w}(n) = \hat{w}(n-1) + g(n)\hat{e}_{n-1}(n)$	
5. $R_\lambda^{-1}(n)$ Updating $R_\lambda^{-1}(n) = \lambda^{-1}\left[R_\lambda^{-1}(n-1) - g(n)\mathbf{x}^T(n)R_\lambda^{-1}(n-1)\right] = \lambda^{-1}\left[R_\lambda^{-1}(n-1) - g(n)\bar{g}^T(n)\right]$	

Table 4 summary of RLS algorithm

3.9 Kalman Filter as equalizer

A complete schematic diagram of adaptive Kalman equalizer is shown in Fig.1. The aim of this equalizer is to reconstruct the original transmitted data at receiver from the received data affected by noise and ISI. Transmitted data is corrupted by the channel parameters. So the modeling of this channel parameters are required.

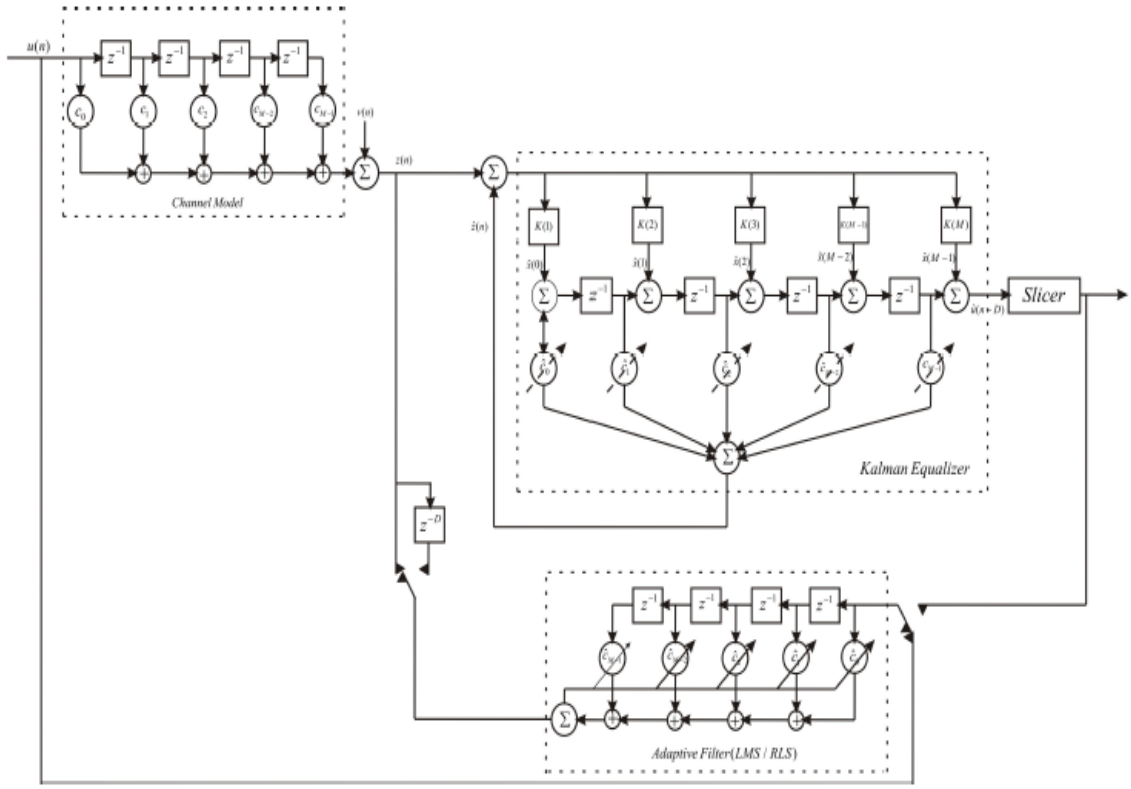


Fig: 3.6 Channel equalization for time varying channel

channel model:-

The communication channel can be modeled by discrete time transversal filter with additive white noise as shown on top left hand side in Figure. 1. A low-pass tapped delay line model of the time varying channel is really nothing more than an finite impulse response (FIR) filter with time varying coefficients. The input-output

description of the FIR filter with time varying coefficients isThe channel output S(n) can be written in terms of channel input x(n) and time varying channel coefficient $h_n[k]$ as

$$\boldsymbol{S[n] = \sum_{k=0}^{q-1} h_n[k]x[n-k]}\text{.....}3.44$$

In presence of noise w[n]

$$\boldsymbol{s[n] + w[n] = \sum_{k=0}^{q-1} h_n[k]x[n-k] + w[n]\text{.....3.45}}$$

Where $w[n]$ is assumed to be white Gaussian noise (WGN) with variance σ^2. It is a slow fading channel so state space model is given by

$$\boldsymbol{h[n] = Ah[n-1] + u[n]\text{.....3.46}}$$

where

$$\boldsymbol{h[n] = [h_n[0], h_n[1], \dots\dots\dots\dots\dots h_n[q-1]]^T\text{.....3.47}}$$

A is a known qxq matrix, u[n] is the vector WGN with covariance matrix Q. A standard assumption that is made to simplify the modeling is that of the uncorrelated scattering [2]. We assume that the tap weights are uncorrelated with each other and hence independent due to jointly Gaussian assumption. As a result, we can let A,Q and Ch, the covariance matrix of h[-1] be diagonal matrices. The vector Gauss-Markov model then becomes q independent scalar models. the measurement model is given by

$$\boldsymbol{y[n] = x^T[n]h[n] + w[n]\text{.....3.48}}$$

We can now form the minimum mean square error (MMSE) estimator for the tapped delay line weights recursively in time using the Kalman filter for this particular problem (vector state and scalar observations).

Equation (3) represents the vector state model and equation (5) is scalar observation or measurement equation. The Kalman filter equations for this problem are

Prediction:

$$\widehat{\boldsymbol{h}}[\boldsymbol{n}/\boldsymbol{n}-\mathbf{1}]=\boldsymbol{A}\widehat{\boldsymbol{h}}[\boldsymbol{n}-\mathbf{1}/\boldsymbol{n}-\mathbf{1}]\ldots\ldots\ldots\ldots\ldots\ldots\mathbf{3.49}$$

Minimum Prediction MSE matrix(qxq):

$$\boldsymbol{M}[\boldsymbol{n}/\boldsymbol{n}-\mathbf{1}]=\boldsymbol{A}\boldsymbol{M}[\boldsymbol{n}/\boldsymbol{n}-\mathbf{1}]\boldsymbol{A}^{\boldsymbol{T}}+\boldsymbol{Q}\ldots\ldots\ldots\ldots\ldots\mathbf{3.50}$$

Kalman Gain:

$$\boldsymbol{K}[\boldsymbol{n}]=\frac{\boldsymbol{M}[\boldsymbol{n}/\boldsymbol{n}-\mathbf{1}]\boldsymbol{x}[\boldsymbol{n}]}{\boldsymbol{\sigma}^{\mathbf{2}}+\boldsymbol{x}^{\boldsymbol{T}}[\boldsymbol{n}]\boldsymbol{M}[\boldsymbol{n}/\boldsymbol{n}-\mathbf{1}]\boldsymbol{x}[\boldsymbol{n}]}\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\mathbf{3.51}$$

Correction:

$$\widehat{\boldsymbol{h}}[\boldsymbol{n}/\boldsymbol{n}]=\widehat{\boldsymbol{h}}[\boldsymbol{n}/\boldsymbol{n}-\mathbf{1}]+\boldsymbol{K}[\boldsymbol{n}](\boldsymbol{y}[\boldsymbol{n}]-\boldsymbol{x}^{\boldsymbol{T}}[\boldsymbol{n}]\widehat{\boldsymbol{h}}[\boldsymbol{n}/\boldsymbol{n}-\mathbf{1}]\ldots\ldots\ldots\ldots\mathbf{3.52}$$

Minimum MSE:

$$\mathbf{M}[\boldsymbol{n}|\boldsymbol{n}]\ =\ (\mathbf{I}-\mathbf{K}[\boldsymbol{n}]\mathbf{X}^{\mathbf{T}}[\mathbf{n}])\mathbf{M}[\boldsymbol{n}|\boldsymbol{n}-\mathbf{1}]\ldots\ldots\ldots\mathbf{3.53}$$

Initialization matrices are

$\hat{\mathbf{h}}[-1\,|\,-1]=\mu_h=0$

$\mathbf{M}[-1\,|\,-1]=C_h=100\mathbf{I}$

3.10 Implementing The Kalman Filter

Although the Kalman filter algorithm itself appears to be quite general and straightforward, its successful implementation tends to be very problem-specific, relying heavily on engineering judgment to adjust and tune process and sensor models. As a rule of thumb, it takes an order of magnitude more time and effort to tune and adjust a filter to work well, over the time it takes to implement the basic algorithm.

This rule increases to two orders of magnitude in the case of non-linear process and observation models. There are two main reasons for this: First, in dealing with real data from a real process it is very rarely the cases that the true state of nature is known. That is, there is never any absolute truth by which to judge the performance of the algorithm, all that is available for this judgment are the observations which are themselves used by the filter. This leads to a problem of introspection in which the only way to judge the performance of a filter is with respect to other possible filter performances. The second reason for the difficulty of implementation is simply not knowing when the filter performance is limited by the adequacy of the sensor and process models rather than an inability to fine tune the algorithm: can I do any better by improving my models or am I simply limited by the performance of the algorithm itself ? Again this is often a question of engineering judgment which can only be based on the comparison of different possible filters with respect to the observations that are made.

As in all matters of engineering judgment there are some well understood procedures and rules which provide a systematic means of approaching the problem of design and implementation. In the case of the Kalman filter these rules are as follows:

1. Understand your sensor: The first step is simply to be familiar with the physics of the device: propagation medium,wave-length or emission characteristics, maximum and minimum ranges, etc. The second step is to acquire as much data as possible, in a variety of situations, from the sensor to be employed. A surprising amount can be learnt by simply looking at this data and appreciating what kind of information is likely to be available to the filter. It is quite pointless designing a filter without knowing what information will be available for use.

2. Understand your process: Again familiarity with the kinematics, physics or geometry of the process under consideration is essential: parameters of importance should be identified, constraints and physical limits made precise, key time-constants should be measured, etc. Observation of the process in a variety of operating modes,

with additional instrumentation if required, can yield surprising insights into filter design and implementation. It (almost) goes without saying that it is simply not sensible to begin designing a filter without a clear understanding of the problem to be solved.

3. Model your sensor: Having obtained as much information as possible from the device, an accurate kinematic and statistical model of the sensor must be developed. In the Kalman filter algorithm this simply reduces to the construction of an appropriate observation model H(k) and noise source v(k). The performance of the filter will be directly dependent on the adequacy of this model. As we have and will continue to stress, there is simply no substitute for developing precise and detailed models of the sensing process.

4. Model your process: The first step is to build as accurate a truth model' as possible, describing all aspects of the process to be estimated. This model will undoubtedly be too large and intractable to be employed directly in the filter algorithm but is still an essential step in understanding which states are significant and which marginal in obtaining desirable filter performance. The second step is to reduce this model to those states which have a direct and significant impact on filter performance and to construct an appropriate process model F(k) and process noise w(k). This has to be done on a case-by-case basis with respect to an overall 'state-budget'.

5. Filter Coding: The easiest part of the implementation is simply coding the Kalman filter algorithm and models generated in the preceding analysis. Some clearly defined rules exist for sequencing the various components of the algorithm and computing of state estimates and their associated covariances.

6. Initialization: The recursive formulation of the Kalman filter algorithm means that we must provide some reasonable guess for the initial conditions $\hat{X}(0 \mid 0)$ and $P(0 \mid 0)$. Although not critical to long-term performance in the linear algorithm (we shall see that the effects of initial conditions diminish rapidly with time) initialization is still of importance in the filtering of data from real systems.

7. Analysis of Innovation Sequence: The first and most important method of analyzing filter performance is using the innovation or residual sequence. Recall that the innovation is simply the difference between the true observation and the predicted observation and that under the assumptions made in deriving the Kalman filter, the innovation sequence will be white and uncorrelated with known covariance S(k). We will see that testing the innovation sequence for these properties tells us a great deal about how the filter is working and can be used directly to tune filter performance.

8. Analysis of Steady-State Performance: The innovation is a single measure of filter performance which is affected by both the observation and process models. The second step in analysing filter performance is to separate out these two factors by looking simultaneously at both the observation sequence and at the steady-state properties of the filter; the state estimates, state predictions, and their respective covariances.

9. Analysis of Error Conditions: In real systems a final step is required to identify and eliminate erroneous or spurious data from consideration in the filtering process; to ensure that the filter is sufficiently robust for proper use.

The object of this section is to provide as much practical advice as possible on how to design and build a linear Kalman filter. The previous section described the problem of building and validating sensor and process models. Here we will concentrate on the implementation of the algorithm and the tuning of the filter to

achieve desired performance. Although a great deal of what follows must be example specific, the techniques and procedures employed can and should be generalized in all applications of the Kalman filter Algorithm.

We begin be describing the overall estimation architecture and the way in which it should be implemented. Taking the example of one-dimensional target motion developed throughout this section we show first how the filter is initialized and then describe how the performance of the filter may be analyzed using the innovation sequence and steady-state conditions.

3.11 Advantages

The single most important advantage of the Kalman Algorithm over the conventional steepest-desect algorithm is its convergence rate. The improvement in the convergence rate is more in the case where the condition number (or the eigen-value spread) of the Covariance matrix is large. This is because the Kalman algorithm utilizes more parameters than the gradient algorithm. The parameters that control the Kalman algorithm are the N elements of the Kalman weight vector KN(t), with each element controlling a tap-weight vector.

3.12 Disadvantages.

The computational complexity of the Kalman algorithm is proportional toN2 and is considerably more when compared to the gradient algorithm. To correct this problem numerous variations of this algorithm have been proposed in literature :

• The Square-Root RLS algorithm proposed by Bierman *[16]*. It has an computation complexity of 1.5N2 + 6.5N.

• Variations of Kalman algorithm that have computational complexities that grow linearly with N have also been proposed in literature [17,1 8, 19].

Another disadvantage of the RLS algorithm is its sensitivity to round-off noise that accumulates due to recursive computations. This may result in instabilities.

4.1 Simulation Results and Discussions

This chapter will present the results of the simulations performed using Matlab, based on the literature review and findings described in earlier chapters. The results were analyzed and presented using the parameters set to evaluate the performance of an adaptive equalizer, namely:

- Channel adaptation
- Tap weight of kalman filter
- Kalman filter gain
- Squared difference of input to Channel and output of Equalizer(MMSE)

These analyzed results will be accompanied by discussions of the observations made compared to the theoretical findings in previous chapters.

Here we are using simple gauss-markov channel due to simplicity in calculation and no. of multipath for the channel is two.

The simulations involved the following:

- Kalman filter as equalizer

4.2 kalman filter as equalizer

Let the Kalman filter estimator have $q = 2$ weights. Assume a state model with

$A=\begin{bmatrix}0.99 & 0\\ 0 & 0.998\end{bmatrix}$ And $Q=\begin{bmatrix}0.0001 & 0\\ 0 & 0.0001\end{bmatrix}$

True values of the weights are shown in Fig. 4.1, in which $h_n[0]$ is decaying to zero while $h_n[1]$ is fairly constant. This is because the mean of the weights will be zero in steady state.

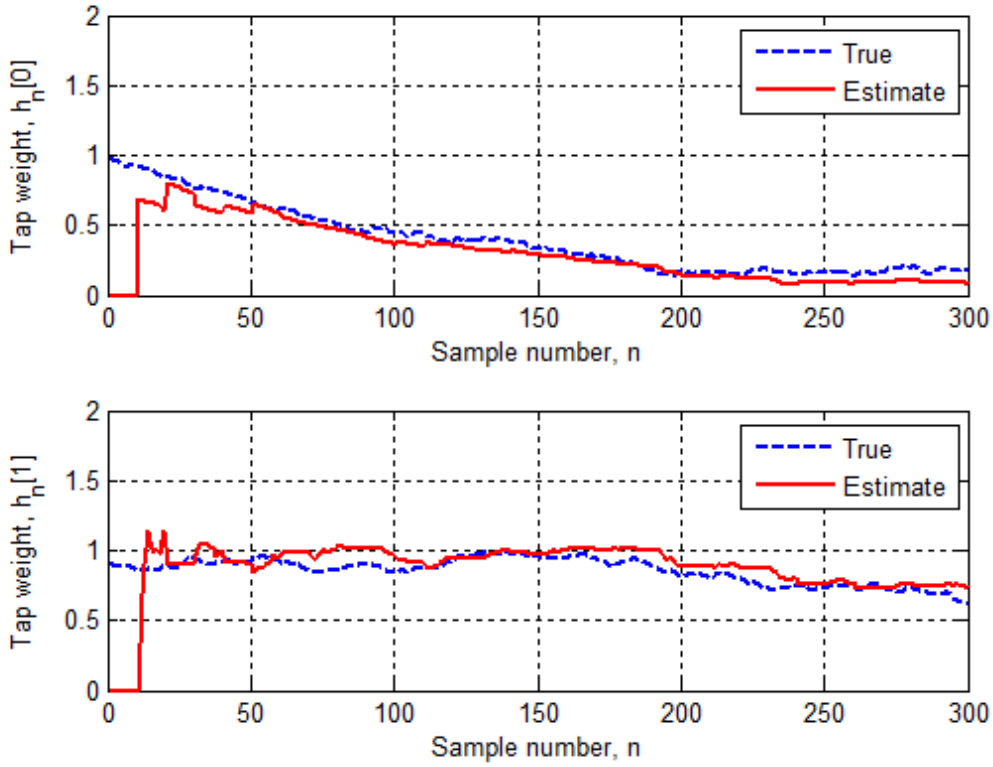


Fig: 4.1 True value and the estimated value of the tap weight of kalman filter

As shown in the figure 4.1 the blue line shows the true value of the filter coefficient (tap weight) and the red line shows the estimated value. Due to the smaller value of $[A]_{11}$, $h_n[0]$ will decay more rapidly. Also, note that the eigen values of A are just the diagonal elements and they are less than 1 in magnitude.

In practice this is seldom known, so that we usually just choose an arbitrary initial state estimate with a large initial MSE matrix to avoid "biasing" the Kalman filter towards that assumed state. The estimated tap weights are shown in figure 4.1 with the

help of the red line After an initial transient the Kalman filter “locks on" to the true weights and tracks them closely.W hich is our prime requirement.

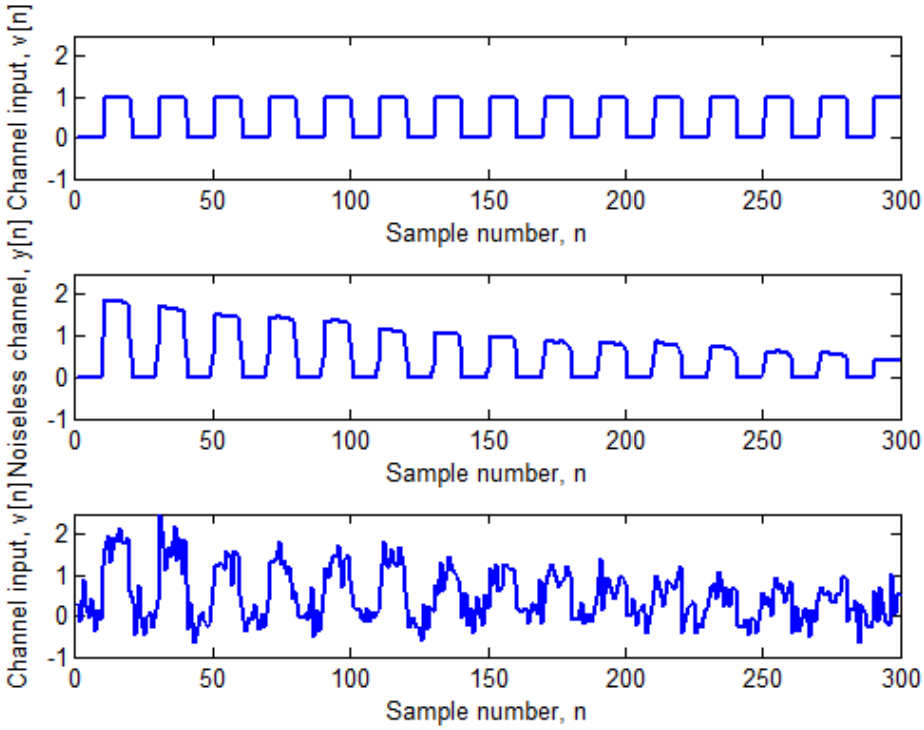


Fig: 4.2 input and out put of channel without noise and with noise

Here figure 4.2 shows the input to the channel with noise and without noise the channel model is gauss-markov model is used due to simplicity.

When observation noise is added with σ= 0.1, Let $\hat{h}$[-1 -1] = = 0 and M[-1 -1] = Ch = 100I, which were chosen to reflect little knowledge about the initial state. In the theoretical development of the Kalman filter the initial state estimate is given by the mean of s[-1].

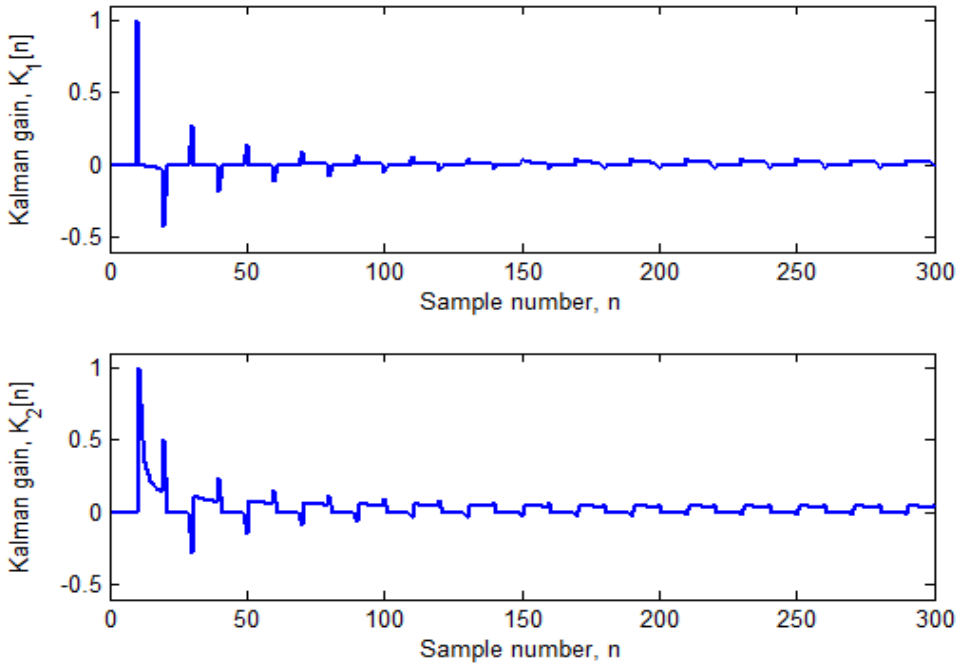


Fig: 4.3 Variation of kalman gain

The Kalman filter gains are shown in figure. 4.2. They appear to attain a periodic steady-state, although this behavior is different than the usual steady-state, since x[n] varies with time and so true. steady-state is never attained. Also, at times the gain is zero, as for example in $[K]_1 = k_1[n]$ for $0 \leq n \leq 4$. This is because at these times the input x[n] is zero and thus the observation contain only noise. The Kalman filter ignores these data samples by forcing the gain to be zero.

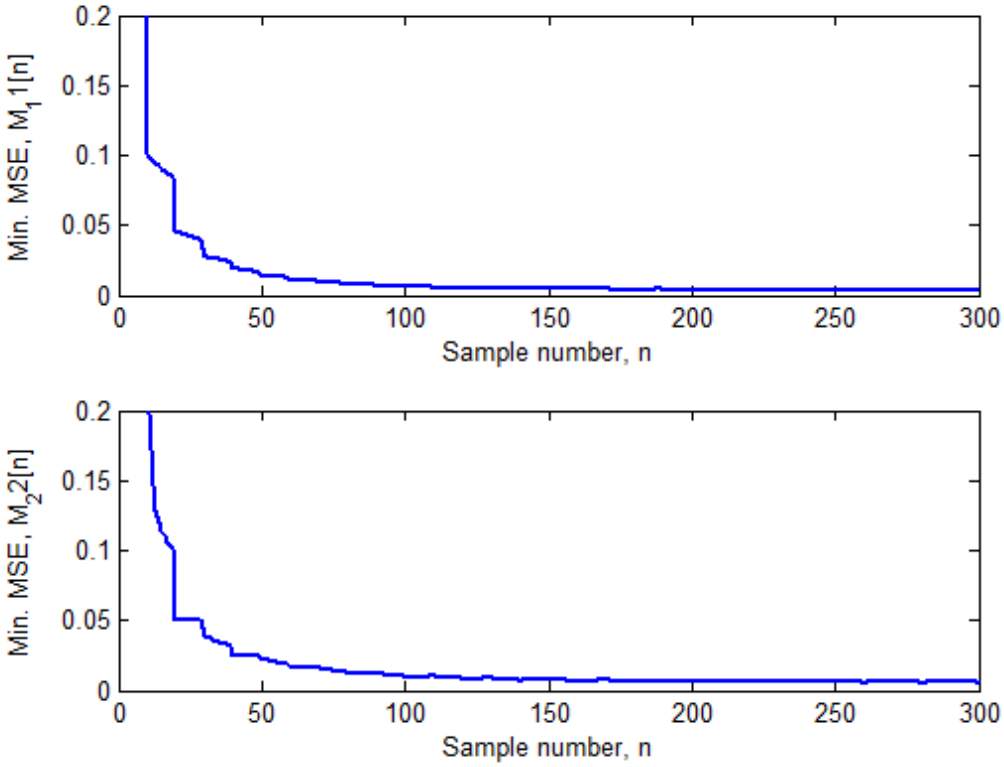


Fig: 4.4 MMSE of the filter

Finally the min. mean square error is shown in the figure 4.3. In the MSE criterion, the tap weight coefficients of the equalizer are adjusted to minimize the mean square values of the error at the output of the equalizer. Error in this sense is the difference between the sent symbol and the equalizer output. Here both ISI and additive noise compose this error. the minimization of MSE results in complete elimination of the ISI, and it will become identical to ZF-equalizer.

5.1 Conclusion

The objective of this thesis is to develop a suitable adaptive equalization technique to mitigate the effects of ISI and noise in a typical communication channel. With the successful development of the adaptive Equalizer and blind Equlization, it offers an excellent alternative to the existing equalization techniques available in the communication industry. This thesis also gives brief introduction of the Blind Equalizer which is a very recent equalizer technique. Here kalman filter is used as adaptive equalizer for the unknown channel.

The adaptive equalizer was implemented using the Recursive Least Square (RLS) technique, using stochastic gradient adaptation, for the direct equalization of the unknown channel. A number of adaptive algorithms had been analyzed and discussed. The RLS algorithm was chosen because of its faster convergence than LMS algorithm and stability. The main contribution of this thesis is the development of a Kalman filter based channel estimation algorithm. We considered a multipath channel with a time varying impulse response. Training sequences are sent periodically to produce snapshot estimates of the channel.

We assumed a Gauss-Markov model for the channel due to its simplicity in calculation. we can now predict the state of the channel (with lesser accuracy of course) without having to wait for the data estimate to arrive. The Kalman estimator improved upon the performance of the data estimator by almost thirty percent on each path. Since the only way (assuming channel noise is not under the operators control) to increase the accuracy of the data estimate is to increase the length of the training sequence, the Kalman estimator provides an efficient technique of improving the channel estimate without wasting any more bandwidth. In radio systems where

bandwidth is prohibitively expensive or there is just no more room on the frame for any more training sequence information, the Kalman estimator solution becomes even more attractive.

In short we can say in this thesis the channel is modeled as an FIR filter with time varying coefficients. The observation model is assumed to be Gauss-Markov for tap weights. Kalman filter is used to estimate the time varying coefficients of the channel.

5.2 Future work

The work presented in this thesis can be extended various ways including the following:

Use of multiple sampling rates[20] : In this thesis it is assumed that data estimates arem available at the end of every frame. The channel is assumed to be a constant for the duration of this frame. The Kalman filter based estimator provides current estimates after processing the data based estimate and hence at the end of each frame received. We can increase the usefulness of the method presented in this thesis by running the Kalman filter at a higher rate than the frame rate. In the intervals that no data estimate has arrived, we can perform only the time-update portion of the Kalmanfilter. When data is received, we perform the measurement update portion of the Kalman algorithm. Estimates can then be made available on as fine a division of the time line as we desire. The second advantage is that the data arrival times need not be uniform. For a-periodically available data we merely perform time updates until a data estimate is received.

Correlated paths: In this work, we have assumed that the multipaths are not correlated. Correlated multipaths can still be tracked using this Kalman algorithm but further work needs to be done to modify the Kalman filter to track correlated paths. A

good starting point is reference *[21]* where a Cholesky decomposition is used to generate correlated multipath waves.

Actual Implementation: This algorithm is very well suited for implementation on a live system. The discrete Kalman filter is well documented as a robust algorithm. It will be very interesting to compare the theoretical and actual performance of the algorithm.

References

1.T.S. Rappaport, Wireless Communications Principles and Practice.Publisher: Prentice Hall press. 1996.

2. William C. Jakes, Microwave mobile communications., Publisher: IEEE Press, 1993.

3.Michel C. Jeruchim, Philip Balaban, K. Sam Shanmugan, Simulation Of Communication Systems. Publisher: Plenum, 1992.

4.Rolnnd Zukunfr, S V E H Uar, And Thomas Magesacher," Blind Adaptation Algorithm For Decision Feedback Equalizers With Fractionally-Spaced Feedforward Fllters", IEEE Transaction,2002.

5.Bakhtiar Qutub Ali," A New Blind Equalization Scheme based on Principle of Minimal Disturbance", king fahd university of petroleum and minerals, May 2004

6.John M Senior,"Optical Fiber Communication" McGraw-Hill Series in Electronics Engineering, Edition 3rd, 2000.

7.J. K. Tugnait, L. Tong, and Z. Ding, "Single-User Channel Estimation and Equalization," IEEE Signal Processing Magazine, vol. 17, no.1, pp. 17-28, May. 2000.

8. Jacob Benesty and Yiteng Huang," Adaptive Signal Processing", Springer, 2003.

9. Bernard Widrow and Samuel D. Stearns," Adaptive Signal Processing," Prentice-Hall, 1985.

10. Simon Haykin," Adaptive Filter Theory," Pearson Education, 2002.

11. Openhem, "Signal and System," Pearson Education, 2002.

12. M.S. Grewal and A.P. Andrews. Kalman Filtering Theory and Practise. Prentice Hall,
1993.

13. R.E. Lawrence and H. Kaufman, "The Kalman Filter for the Equalization of a Digital Communications Channel," IEEE Trans. Comm. Tech. (Special Issue on Signal Process. For Digital Comm.), vol. COM-19, pp. 1137-1141, Dec. 1971.

14. Godard, D. "Channel equalization using a Kalman filter for fast data transmission," IBM. J. Res . Develop.,vol. 18, pp.267-273, May 1974.

15. Kamran Azadet Erich F. Haratsch, Helen Kim," Equalization and FEC Techniques for Optical Transceivers", IEEE journal of solid-state circuits, vol. 37, no. 3, March 2002.

16. G.J.Biermann, "Factorization Methods for Sequential Estimation",Academic, New York.

17. G. Carayannis, D.Manolakis, N. Kalouprtsidis, "A fast sequential algorithm for least squares filtering and prediction", IEEE Transactions on Accoustics, Speech and Signal Processing, Vol. ASSP-31, pp. 1394-1402, December 1983.

18. J.M.Cioffi and T.Kailath, "Fast Recursive Least Squares Transversal Filters for Adaptive Filtering", IEEE Transactions on Accoustics, Speech and Signal Processing, Vol. ASSP-32, pp.304-337, April 1984.

19. D.T.M. Slock and T.Kailath, "Numerically Stable Fast Recursive Least Squares Transversal Filters", Proc. Int. Conf. Accoustics, Speech and Signal Processing, pp. 1365-1368, New York, April 1988.

20. Frank L. Lewis, "Optimal estimation: with an introduction to stochastic control theory." Publisher: Wiley, c1986.

21. T. Hattori, K. Hirade, "Generation Method Of Mutually Correlated Multipath Fading waves," Electronics and Communications in Japan, Vol. 59-B, pp.69-76, 1976.