NEAR EAST UNIVERSITY

GRADUATE SCHOOL OF APPLIED SCIENCES

ADAPTIVE EQUALIZATION FOR MULTIPATH RAYLEIGH FADING CHANNELS

İlke Uludağ

Master's Thesis

Department of Electrical and Electronic Engineering

Nicosia – 2007

ACKNOWLEDGEMENTS

I would like to acknowledge the support and guidance from my advisor, Prof. Dr. Fahreddin Sadıkoğlu, whose suggestions and directions have a major influence on all aspects of my thesis. I would also like to thank Assoc. Prof. Dr. Adnan Khashman for his encouragement throughout my difficult times. I would like to thank all my friends in EE department for making the past two and a half years at Near East University, a wonderful and memorable experience. Special thanks to Asst. Prof. Dr. Hüseyin Sevay. I am also grateful to Mr. Ahmet Zaim for helping me draw equalizer structures in MS Visio. Finally, great thanks to my parents for their tremendous support, love, and guidance in life. This thesis is dedicated to them.

ABSTRACT

Intersymbol interference (ISI) caused by multipath propagation is a significant problem in any digital mobile communication system. The channel characteristics are also time varying due to the movement of the mobile station relative to its surrounding. In a system such as GSM, the receiver must be able to estimate the channel and compensate for channel distortion adaptively. In a conventional GSM receiver, this task is implemented by a Maximum Likelihood Sequence Estimation (MLSE) equalizer using the Viterbi algorithm. This thesis models a multipath Rayleigh fading channel for the mobile environment and designs a frequency domain adaptive equalizer that implements the Least Mean Square (LMS) algorithm as a successful alternative to the traditional method.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ISI	Intersymbol Interference
BER	Bit Error Rate
SNR	Signal to Noise Ratio
FIR	Finite Impulse Response
IIR	Infinite Impulse Response
LMS	Least Mean Square
DFE	Decision Feedback Equalizer
FTF	Fast Transversal Filter
RLS	Recursive Least Squares
TDMA	Time Division Multiple Access
FDMA	Frequency Division Multiple Access
CMA	Constant Modulus Algorithm
SCORE	Spectral Coherence Restoral Algorithm
MLSE	Maximum Likelihood Sequence Estimation
LTE	Linear Transversal Equalizer
FFF	Feed forward Filter
FBF	Feedback Filter
MSE	Mean Square Error
GSM	Global System for Mobile Communications
FDAE	Frequency Domain Adaptive Equalizer
DSP	Digital Signal Processing
GMSK	Gaussian Minimum Shift Keying

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Performance degradation in a mobile radio communication system is due to physical phenomena, such as multipath fading, time and Doppler delay spread that produce intersymbol interference. To counteract these impairments, the mobile receiver such as the one in a GSM handset, uses a maximum likelihood sequence estimation (MLSE) equalizer based on the Viterbi algorithm. This algorithm is well known to be the optimum solution for detecting an information sequence corrupted by ISI. Its optimality is based on the assumption that the statistical behavior of the channel is known. When this assumption does not hold, as in mobile communications applications where the environment changes not only for each different connection but also within the same call, adaptive equalization is needed to derive the correct parameters of the fading channel so that the fading channel effect can be reversed.

In this thesis, a different approach other than using the MLSE equalizer is taken to design a frequency domain adaptive equalizer with the LMS Gradient algorithm for the GSM mobile radio communication system. The proposed adaptive filter uses a pre-defined preamble training sequence for adaptation. In order to test the performance of the proposed adaptive equalizer, a Rayleigh mutipath fading channel with Doppler spread is implemented as well. Both the channel model and the equalizer are implemented using the built-in functions in MATLAB package.

The thesis is organized as follows. Chapter 1 explains the problem of intersymbol interference. Chapter 2 explains mobile radio propagation and its characteristics. Chapter 3 provides, in detail, different equalizer types and algorithms implemented. Chapter 4 models a Rayleigh multipath fading channel for mobile environment and designs a frequency domain adaptive equalizer (FDAE) to minimize ISI created by the channel. Chapter 5 presents test results and discussion of the designed adaptive equalizer. Finally, conclusions and some recommendations for future work are presented.

1. INTERSYMBOL INTERFERENCE

1.1 Overview

In mobile communications systems, there has been a growing demand for high data rate services such as video phone, high-quality digital distribution of music, and digital television terrestrial broadcasting (DTTB). In such systems, the delay spread of the channel becomes a major impairment to cope with, since it may cause severe intersymbol interference (ISI) due to multipath propagation. In this chapter, the problem of intersymbol interference will be explained.

1.2 Problem Statement

Intersymbol interference (ISI) is an unavoidable consequence of both wired and wireless communication systems. Figure 1.1 shows a data sequence, [1 0 1 1 0], which is to be transmitted.

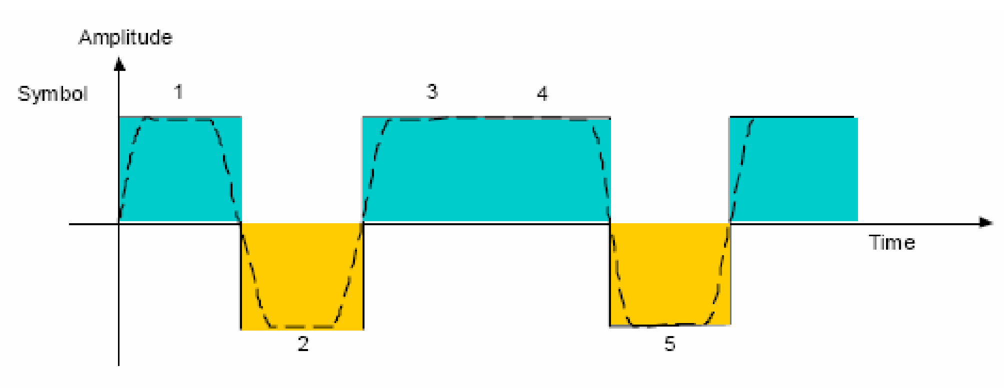


Figure 1.1 Transmitted signal [11]

This sequence is in form of square pulses. Square pulses are fine as an abstraction but in practice they are hard to create and also require far too much bandwidth so they are shaped as dotted lines as shown above.

The shaped version looks essentially like a square pulse and even visually, one can easily see what was transmitted. Figure 1.2 shows each symbol as it is received. A tail of energy, created by the transmission medium, can be observed that lasts much longer than intended. The energy from symbols 1 and 2 goes all the way into symbol 3. Each symbol interferes with one or more of the subsequent symbols. The circled areas show areas of large interference [7].

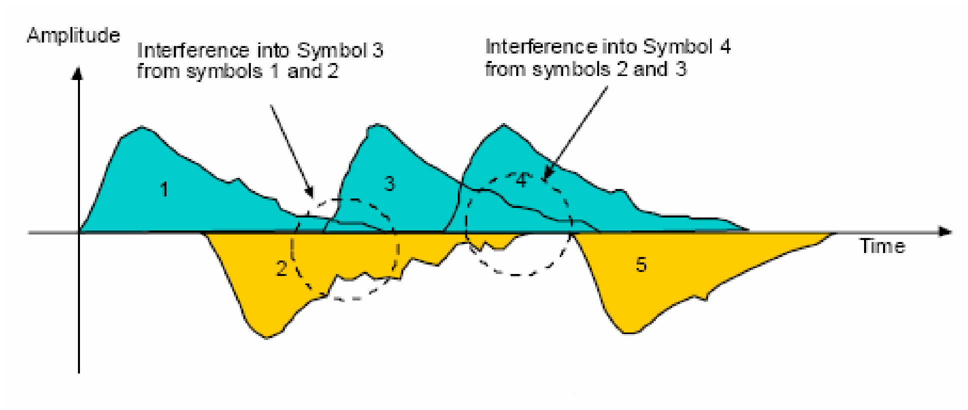


Figure 1.2 Received signal [11]

Figure 1.3 shows the actual signal seen by the receiver. It is the sum of all these distorted symbols.

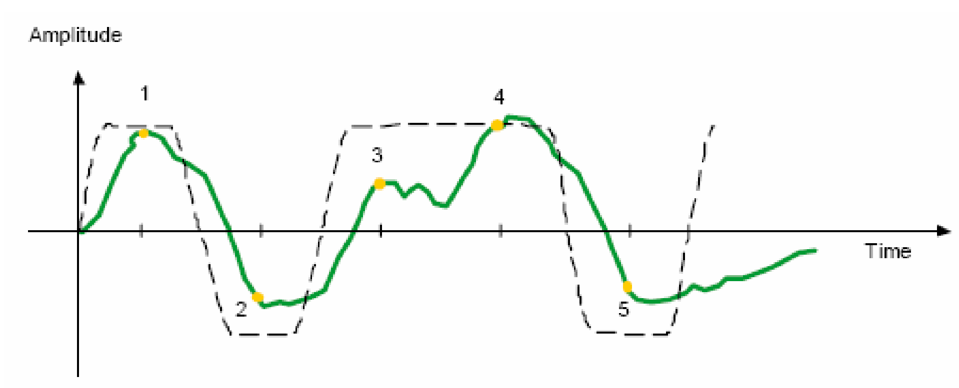


Figure 1.3 Received signal vs. transmitted signal [11]

Compared to the transmitted dashed line signal, the received signal looks quite indistinct. The receiver does not actually see this signal; it sees only the little dots, the value of the amplitude at the timing instant. For symbol 3, this value is approximately half of the transmitted value which makes this particular symbol more susceptible to noise and incorrect interpretation. This phenomenon is the result of the symbol delay and smearing. This spreading and smearing of symbols such that the energy from one symbol affects the next in such a way that the received signal has a higher probability of being interpreted incorrectly is called intersymbol interference (ISI).

ISI can be caused by many different reasons. It can be caused by filtering effects from hardware or frequency selective fading, from non-linearities and from charging effects. Very few systems are immune from it and it is nearly always present in wireless communications [17].

1.3 Equalization to Combat Intersymbol Interference

The main problem is that energy, which is confined to one symbol, leaks into others so one of the simplest things that can be done to reduce ISI is to just slow down the signal, that is to say "transmit the next pulse of information only after allowing the received signal has to damp down". The time it takes for the signal to die down is called delay spread, whereas the original time of the pulse is called the symbol time. If delay spread is less than or equal to the symbol time then no ISI will result, otherwise ISI occurs. Slowing down the data rate is an easy but an unacceptable solution. The optimal method to counter ISI that does not require reducing the bit rate is to adaptively equalize the channel. This is accomplished by using an adaptive equalizer.

1.4 Summary

In this chapter, the problem of intersymbol interference and delay spread were explained. In addition, the need for an optimum solution to the problem was stated. In the following chapter, the mobile radio propagation details will be given.

2. MOBILE RADIO PROPAGATION

2.1 Overview

In chapter 1, ISI concept was introduced and details of the problem were given. Building onto chapter 1, in this chapter, time varying multipath fading channel and its characteristics will be explained. In addition, Rayleigh and Rician multipath fading channels and Doppler spread concept will be given. Fast and slow fading channel phenomena will be established by correlating the coherence time with the symbol time.

2.2 Time Varying Multipath Fading Channel

Fundamentally, mobile radio communication channels are time varying, multipath fading channels. In a radio communication system, there are many paths for a signal to travel from a transmitter to a receiver. Sometimes there is a direct path where the signal travels without being obstructed. In most cases, components of the signal are reflected by the ground and objects between the transmitter and the receiver such as buildings, vehicles, and hills or refracted by different atmospheric layers. These components travel in different paths and merge at the receiver. Figure 2.1 illustrates this phenomenon [4].

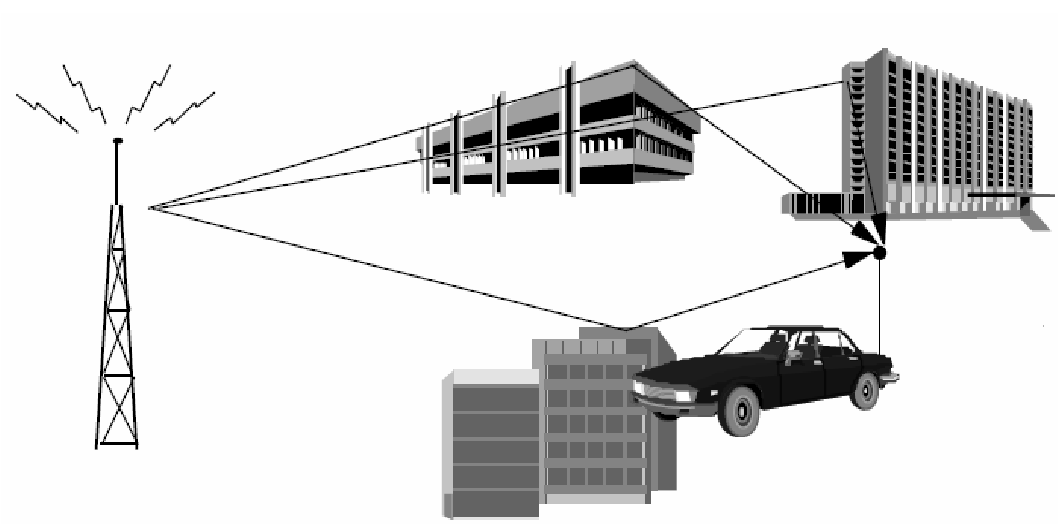


Figure 2.1 Multipath fading channel [19]

Each path has a different physical length. Thus, signals on each path suffer different transmission delays due to the finite propagation velocity. The superposition of these signals at the receiver results in destructive or constructive interference, depending on the relative delays involved. The fact that the environment changes as time passes leads to signal variation. This is called time variant. Signals are also influenced by the motion of a terminal. A short distance movement can cause an apparent change in the propagation paths and in turn the strength of the received signals.

2.3 Multipath Fading Channel Characteristics

Both the propagation delays and the attenuation factors are time variant as a result of changes in the structure of the medium. The received bandpass signal may be expressed in the form

$$x(t) = \sum_{n} \alpha_n(t) s(t - \tau_n(t)) \qquad (2.1)$$

where $s(t)$ is the transmitted signal, $\alpha_n(t)$ is the attenuation factor for the signal received on the nth path and $\tau_n(t)$ is the propagation delay for the nth path. $s(t)$ can be expressed as

$$s(t) = \mathrm{Re}\left\lfloor s_l(t) e^{j2\Pi f_c t} \right\rfloor \qquad (2.2)$$

where $s_l(t)$ is the equivalent lowpass transmitted signal. Substituting (2.2) into (2.1) yields

$$x(t) = \mathrm{Re}\left\{ \left[\sum_{n} \alpha_n(t) e^{-j2\Pi f_c \tau_n(t)} s_l(t - \tau_n(t)) \right] e^{j2\Pi f_c t} \right\} \qquad (2.3)$$

From Equation (2.3), the equivalent lowpass received signal is

$$r_l(t) = \sum_n \alpha_n(t) e^{-j2\Pi f_c \tau_n(t)} s_l(t - \tau_n(t)) \qquad (2.4)$$

It follows that the equivalent lowpass channel is described by the time variant impulse response

$$c(\tau;t) = \sum_n \alpha_n(t) e^{-j2\Pi f_c \tau_n(t)} \delta(\tau - \tau_n(t)) \qquad (2.5)$$

$c(\tau;t)$ represents the response of the channel at time t due to an impulse applied at time $t-\tau$. When a large number of propagation paths exist, $c(\tau;t)$ can be modeled as a complex-valued Gaussian random process [5]. Thus, the envelope $|c(\tau;t)|$ at any instant t is Rayleigh-distributed, as shown in Figure 2.2.

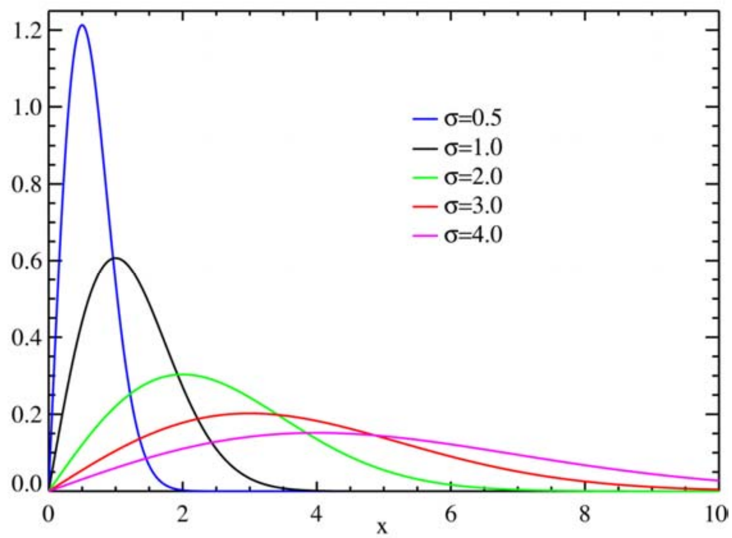


Figure 2.2 Rayleigh distribution [7]

$$f(x) = \begin{cases} \frac{x}{\sigma^2} \exp\left(-\frac{x^2}{2\sigma^2}\right) & \text{for } 0 \le x \le \infty \\ 0 & \text{for } x < 0 \end{cases} \qquad (2.6)$$

In this case, the channel is said to be a Rayleigh fading channel. In the event that there are fixed scatterers or signal refectors in the medium, in addition to randomly moving scatterers, $c(\tau;t)$ can no longer be modeled as having zero mean. In this case, the envelope $|c(\tau;t)|$ has a Rician distribution, as shown in Figure 2.3.

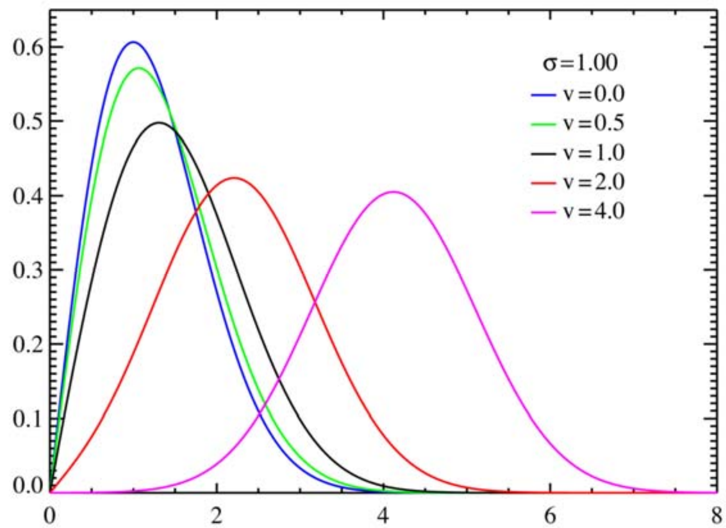


Figure 2.3 Rician distribution [7]

$$f(x) = \begin{cases} \dfrac{x}{\sigma^2}\exp\left(-\dfrac{\left(x^2 - v^2\right)}{2\sigma^2}\right) I_0\left(\dfrac{xv}{\sigma^2}\right) & \text{for } v \geq 0, x \geq 0 \\ 0 & \text{for } x < 0 \end{cases} \tag{2.7}$$

In this case, the channel is said to be a Rician fading channel. Assuming that $c(\tau;t)$ is wide-sense-stationary (WSS), the autocorrelation function of $c(\tau;t)$ can be defined as

$$\phi_c(\tau_1, \tau_2; \Delta t) = \frac{1}{2} E\left[c^*(\tau_1; t) c(\tau_2; t + \Delta t)\right] \tag{2.8}$$

where * defines conjugate. In most radio transmission media, the attenuation and phase shift of the channel associated with path delay τ_1 is uncorrelated with the attenuation and phase shift associated with path delay τ_2. This is generally named as uncorrelated scattering. Under this assumption,

$$\phi_c(\tau_1, \tau_2; \Delta t) = \phi_c(\tau_1; \Delta t)\delta(\tau_1 - \tau_2) \tag{2.9}$$

$\phi_c(\tau) = \phi_c(\tau;0)$ is simply the average power output of the channel as a function of the time delay τ. For this reason, $\phi_c(\tau)$ is called the multipath intensity profile or the delay power spectrum of the channel. The range of values of τ over which $\phi_c(\tau)$ is essentially nonzero is called the multipath spread or delay spread of the channel and is denoted by T_m.

The time variant transfer function $C(f;t)$ can be defined as the Fourier transform of $c(\tau;t)$. That is,

$$C(f;t) = \int_{-\infty}^{\infty} c(\tau;t) e^{-j2\Pi f\tau} d\tau \tag{2.10}$$

If $c(\tau;t)$ is modeled as a complex-valued zero-mean Gaussian random process in the t variable, then it follows that $C(f;t)$ also has the same statistics. Under the assumption that the channel is WSS, the autocorrelation function $C(f;t)$ is the Fourier transform of the multipath intensity profile, i.e.,

$$\phi_C(\Delta f;\Delta t) = \phi_C(f_1, f_2;\Delta t) = \Im\{\phi_c(\tau;\Delta t)\} \tag{2.11}$$

where $\Delta f = f_2 - f_1$. $\phi_c(\Delta f;\Delta t)$ is called the spaced-frequency, spaced-time correlation function of the channel [5]. If $\Delta t = 0$, then

$$\phi_C(\Delta f) = \Im\{\phi_c(\tau)\} \tag{2.12}$$

As a result of the Fourier transform relationship between $\phi_C(\Delta f)$ and $\phi_c(\tau)$, the reciprocal of the multipath spread is a measure of the coherent bandwidth $(\Delta f)_c$ of the channel. That is,

$$(\Delta f)_c \approx \frac{1}{T_m} \tag{2.13}$$

If $(\Delta f)_c$ is small compared to the bandwidth of the transmitted signal, then the channel is said to be frequency-selective. In this case, the signal is severely distorted by the channel. On the other hand, if $(\Delta f)_c$ is large in comparison with the bandwidth of the transmitted signal, then the channel is said to be frequency-nonselective.

If the Fourier transform of $\phi_C(\Delta f;\Delta t)$ is defined with respect to the variable Δt to be the function $S_C(\Delta f;\lambda)$ with $\Delta f = 0$, then the relation becomes

$$S_C(\lambda) = \int_{-\infty}^{\infty} \phi_C(0;\Delta t) e^{-j2\Pi\lambda\Delta t} d\Delta t \qquad (2.14)$$

The function $S_C(\lambda)$ is a power spectrum that gives the signal intensity as a function of the Doppler frequency λ. Hence, $S_C(\lambda)$ is named as the Doppler spectrum of the channel. The range of values of λ over which $S_C(\lambda)$ is essentially nonzero is called the Doppler spread B_d of the channel. Due to the Fourier transform relationship between $S_C(\lambda)$ and $\phi_C(\Delta t)$, the reciprocal of B_d is a measure of the coherence time $(\Delta t)_c$ of the channel. That is,

$$(\Delta t)_c = \frac{1}{B_d} \qquad (2.15)$$

If the coherence time is larger than the symbol period, the channel is said to be a slow-fading channel. On the other hand, if the coherence time is smaller than the symbol period, the channel is a fast-fading channel [6].

2.4 Summary

In this chapter, fundamentals of mobile radio propagation were summarized. Rayleigh and Rician multipath fading channels, Doppler frequency, fast and slow fading channels were explained. The correlation between the coherence time and the symbol time was shown. Out of this correlation, the slow and fast fading channel concepts were explained.

3. ADAPTIVE EQUALIZERS

3.1 Overview

All wireless communication systems demand signal processing methods that are intended to improve the link performance in hostile mobile radio environments. As seen in chapter 2, the mobile radio channel is particularly dynamic due to multipath propagation and Doppler shift. These effects have a strong negative impact on the bit error rate (BER) of any modulation technique. Mobile radio channel impairments significantly distort or fade the signal at the receiver. Equalization is a technique which can be used to improve received signal quality and link performance.

Equalization compensates for intersymbol interference (ISI) created by multipath within time dispersive channels. An equalizer within a receiver compensates for the average range of expected channel amplitude and delay characteristics. Equalizers have to be adaptive because the channel characteristics are generally unknown and time varying. In this chapter, equalization basics, different equalizer types and algorithms will be presented.

3.2 Equalization Basics

Intersymbol interference (ISI) caused by multipath in bandlimited (frequency selective) time dispersive channels distorts the transmitted signal, resulting in bit errors at the receiver. ISI has been identified as the major obstacle to high-speed data transmission over wireless channels. Equalization is a method developed to combat intersymbol interference.

In general, the term equalization describes any signal processing operation that minimizes ISI. In radio channels, a number of adaptive equalizers can be used to cancel interference while providing diversity. Since the mobile fading channel is random and time varying, equalizers must track the time varying characteristics of the mobile channel continuously, and because of this, they are called adaptive equalizers.

The general operating modes of an adaptive equalizer include training and tracking. First, a predefined, fixed-length training sequence is sent by the transmitter so that the receiver's equalizer may adapt to a proper setting for minimum bit error rate

(BER) detection. The training sequence is typically a pseudorandom binary signal or a fixed, prescribed bit pattern. Immediately following this training sequence, the user data (which may or may not include coding bits) is sent, and the adaptive equalizer at the receiver utilizes a recursive algorithm to evaluate the channel and estimate filter coefficients to compensate for the distortion created by multipath in the channel. The training sequence is designed to permit an equalizer at the receiver to acquire the proper filter coefficients in the worst possible channel conditions (e.g., fastest velocity, longest time delay spread, deepest fades, etc.) so that when the training sequence is finished, the filter coefficients are near the optimal values for reception of user data. As user data are received, the adaptive algorithm of the equalizer tracks the changing channel. As a consequence, the adaptive equalizer is continually changing its filter characteristics over time. When an equalizer has been properly trained, it is said to have converged.

The timespan over which an equalizer converges is a function of the equalizer algorithm, the equalizer structure, and the time rate of change of the multipath radio channel. Equalizers require periodic retraining in order to maintain effective ISI cancellation, and are commonly used in digital communication systems where user data is segmented into short time blocks or time slots. Time division multiple access (TDMA) wireless systems are particularly well suited for equalizers. TDMA systems send data in fixed-length time blocks, and the training sequence is usually sent at the beginning of a block. Each time a new data block is received, the equalizer is retrained using the same training sequence [2].

An equalizer is usually implemented at baseband or at IF in a receiver. Since the baseband complex envelope can be used to represent bandpass waveforms, the channel response, demodulated signal, and adaptive equalizer algorithms are usually simulated and implemented at baseband.

Figure 3.1 shows a block diagram of a communications system with an adaptive equalizer at the receiver.

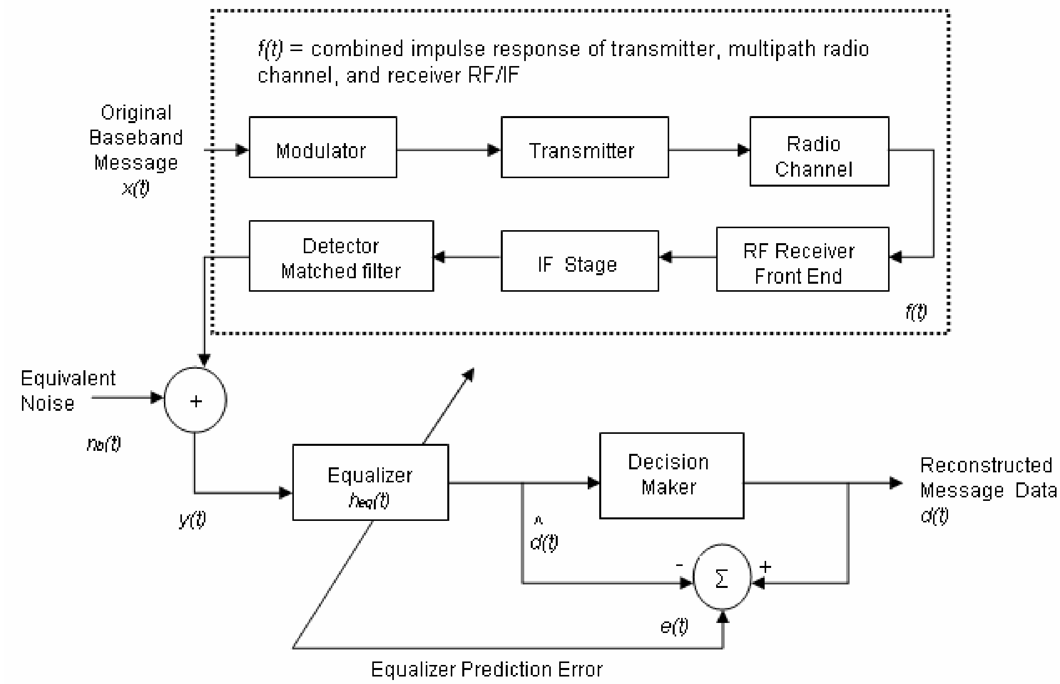


Figure 3.1 Block diagram of a simplified communications system using an adaptive equalizer at the receiver [1]

If $x(t)$ is the original information signal, and $f(t)$ is the combined complex baseband impulse response of the transmitter, channel, and the RF/IF sections of the receiver, the signal received by the equalizer may be expressed as

$$y(t) = x(t) \otimes f^*(t) + n_b(t) \tag{3.1}$$

where $f^*(t)$ denotes the complex conjugate of $f(t)$, $n_b(t)$ is the baseband noise at the input of the equalizer, and $\otimes$ denotes the convolution operation. If the impulse response of the equalizer is $h_{eq}(t)$, then the output of the equalizer is

$$\begin{aligned} d(t) &= x(t) \otimes f^*(t) \otimes h_{eq}(t) + n_b(t) \otimes h_{eq}(t) \qquad (3.2) \\ &= x(t) \otimes g(t) + n_b(t) \otimes h_{eq}(t) \end{aligned}$$

where $g(t)$ is the combined impulse response of the transmitter, channel, RF/IF sections of the receiver, and the equalizer at the receiver. The complex baseband impulse response of a transversal filter equalizer is given by

$$h_{eq}(t) = \sum_n c_n \delta(t - nT) \tag{3.3}$$

where c_n are the complex filter coefficients of the equalizer. The desired output of the equalizer is $x(t)$, the original source data. If it is assumed that $n_b(t) = 0$, then in order to force $d(t) = x(t)$ in Equation (3.2), $g(t)$ must be equal to

$$g(t) = f^*(t) \otimes h_{eq}(t) = \delta(t) \tag{3.4}$$

The goal of equalization is to satisfy Equation (3.4) so that the combination of the transmitter, channel, and receiver appear to be an all-pass channel [1]. In the frequency domain, Equation (3.4) can be expressed as

$$H_{eq}(f)F^*(-f) = 1 \tag{3.5}$$

where $H_{eq}(f)$ and $F(f)$ are Fourier transforms of $h_{eq}(t)$ and $f(t)$, respectively.

Equation (3.5) indicates that an equalizer is actually an inverse filter of the channel. If the channel is frequency selective, the equalizer enhances the frequency components with small amplitudes and attenuates the strong frequencies in the received frequency spectrum in order to provide a flat, composite, received frequency response and linear phase response. For a time-varying channel, an adaptive equalizer is designed to track the channel variations so that Equation (3.5) is approximately satisfied.

3.3 Training A Generic Adaptive Equalizer

An adaptive equalizer is a time-varying filter which must constantly be retuned. The basic structure of an adaptive equalizer is shown in Figure 3.2, where the subscript k is used to denote a discrete time index.

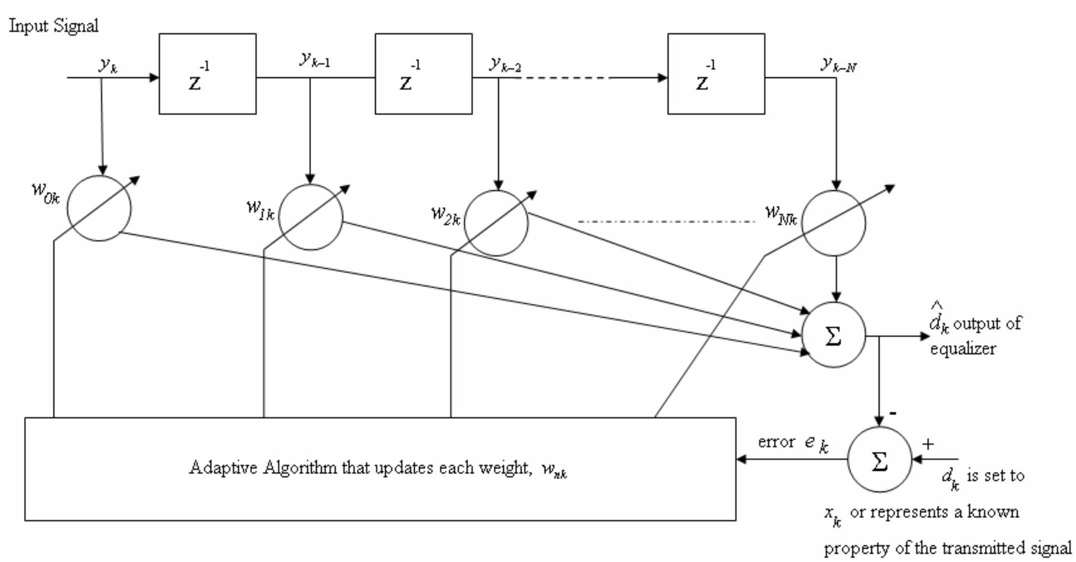


Figure 3.2 A basic linear equalizer during training [1]

It is noticed in Figure 3.2 that there is a single input y_k into the equalizer at any time instant. The value of y_k depends upon the instantaneous state of the radio channel and the specific value of the noise (see Figure 3.1). As such, y_k is a random process. The adaptive equalizer structure shown above is called a transversal filter, and in this case has N delay elements, $N+1$ taps, and $N+1$ tunable complex multipliers, called weights. The weights of the filter are described by their physical location in the delay line structure, and have a second subscript, k, to explicitly show they vary with time. These weights are updated continuously by the adaptive algorithm, either on a sample by sample basis (i.e., whenever k is incremented by one) or on a block by block basis (i.e., whenever a specified number of samples have been clocked into the equalizer).

The adaptive algorithm is controlled by the error signal e_k. This error signal is derived by comparing the output of the equalizer, d_k, with some signal d_k which is either an exact scaled replica of the transmitted signal x_k or which represents a known property of the transmitted signal [13]. The adaptive algorithm uses e_k to minimize a cost function and updates the equalizer weights in a manner that iteratively reduces the cost function. For example, the least mean square (LMS) algorithm searches for the optimum or near-optimum filter weights by performing the following iterative operation:

New weights = Previous weights + (constant) x (Previous error) x (Current input vector) (3.6a)

where

Previous error = Previous desired output – Previous actual output (3.6b)

and the constant may be adjusted by the algorithm to control the variation between filter weights on successive iterations. This process is repeated rapidly in a programming loop while the equalizer attempts to converge, and many techniques (such as gradient or steepest descent algorithms) may be used to minimize the error [22]. Upon reaching convergence, the adaptive algorithm freezes the filter weights until the error signal reaches an acceptable level or until a new training sequence is sent.

Based on classical equalization theory, the most common cost function is the mean square error (MSE) between the desired signal and the output of the equalizer. The MSE is denoted by $E[e(k)e^*(k)]$, and a known training sequence must be periodically transmitted when a replica of the transmitted signal is required at the output of the equalizer (i.e., when d_k is set equal to x_k and is known a priori). By detecting the training sequence, the adaptive algorithm in the receiver is able to compute and minimize the cost function by driving the tap weights until the next training sequence is sent.

A more recent class of adaptive algorithms are able to exploit characteristics of the transmitted signal and do not require training sequences [20]. These modern algorithms are able to acquire equalization through property restoral techniques of the transmitted signal, and are called blind algorithms because they provide equalizer convergence without burdening the transmitter with training overhead. These techniques include algorithms such as the constant modulus algorithm (CMA) and the spectral coherence restoral algorithm (SCORE). CMA is used for constant envelope modulation, and forces the equalizer weights to maintain a constant envelope on the transmitted signal [18].

To study the adaptive equalizer of Figure 3.2, it is helpful to use vector and matrix algebra. Define the input signal to the equalizer as a vector y_k where

$$y_k = [y_k \quad y_{k-1} \quad y_{k-2} \quad \dots \quad y_{k-N}]^T \tag{3.7}$$

It should be clear that the output of the adaptive equalizer is a scalar given by

$$\hat{d}_k = \sum_{n=0}^{N} w_{nk} y_{k-n} \tag{3.8}$$

and following Equation (3.7) a weight vector can be written as

$$w_k = [w_{0k} \quad w_{1\mathrm{k}} \quad w_{2\mathrm{k}} \quad \dots \quad w_{\mathrm{Nk}}]^T \tag{3.9}$$

Using Equations (3.7) and (3.9), Equation (3.8) may be written in vector notation as

$$\hat{d}_k = y_k^T w_k = w_k^T y_k \tag{3.10}$$

It follows that when the desired equalizer output is known (i.e., $d_k = x_k$), the error signal e_k is given by

$$e_k = d_k - \hat{d}_k = x_k - \hat{d}_k \tag{3.11}$$

and from Equation (3.10)

$$e_k = d_k - \hat{d}_k = x_k - \hat{d}_k = x_k - y_k^T w_k = x_k - w_k^T y_k \tag{3.12}$$

To compute the mean square error $|e_k|^2$ at time instant k, Equation (3.12) is squared to obtain

$$|e_k|^2 = x_k^2 + w_k^T y_k y_k^T w_k - 2x_k y_k^T w_k \tag{3.13}$$

Taking the expected value of $|e_k|^2$ over k (which in practice amounts to computing a time average) yields

$$E\left[|e_k|^2\right] = E\left[x_k^2\right] + w_k^T E[y_k y_k^T] w_k - 2E[x_k y_k^T] w_k \tag{3.14}$$

3.4 Equalization Techniques

Equalization techniques can be subdivided into two general categories – linear and nonlinear equalization. These categories are determined from how the output of an adaptive equalizer is used for subsequent control (feedback) of the equalizer [21]. In general, the analog signal $\hat{d}(t)$ is processed by the decision making device in the receiver. The decision maker determines the value of the digital data bit being received and applies a slicing or thresholding operation (a nonlinear operation) in order to determine the value of $d(t)$ (see Figure 3.1). If $d(t)$ is not used in the feedback path to adapt the equalizer, the equalization is linear. On the other hand, if $d(t)$ is fed back to change the subsequent outputs of the equalizer, the equalization is nonlinear. Many filter structures are used to implement linear and nonlinear equalizers. Further, for each structure, there are numerous algorithms used to adapt the equalizer. Figure 3.3 provides a general categorization of the equalization techniques according to the types, structures, and algorithms used.

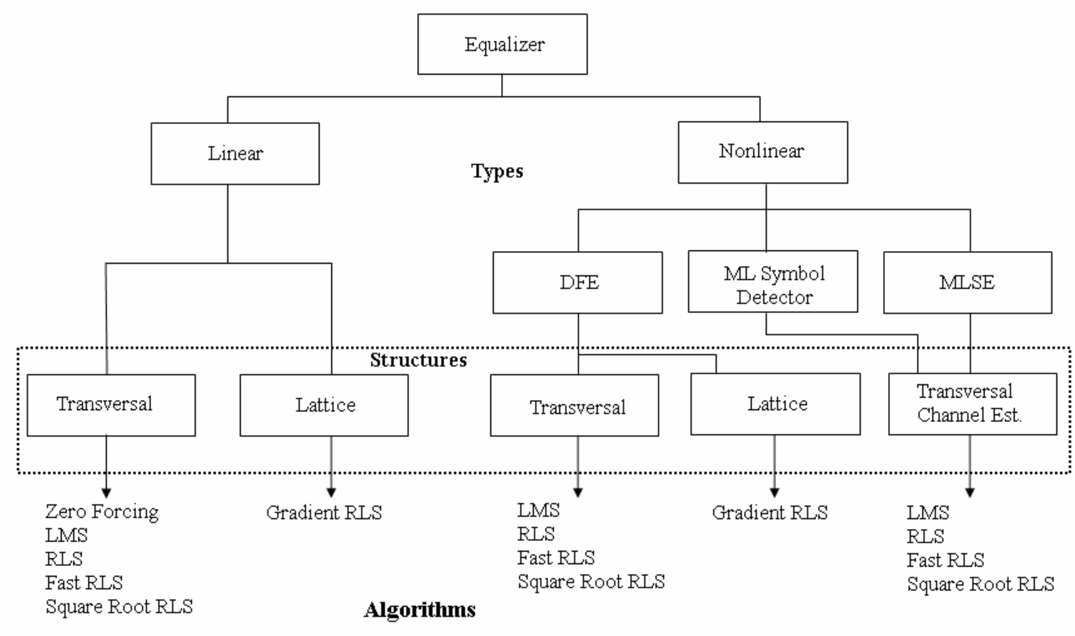


Figure 3.3 Classification of equalizers [1]

The most common equalizer structure is a linear transversal equalizer (LTE). A linear transversal filter is made up of tapped delay lines, with the tappings spaced a symbol period (T_s) apart, as shown in Figure 3.4.

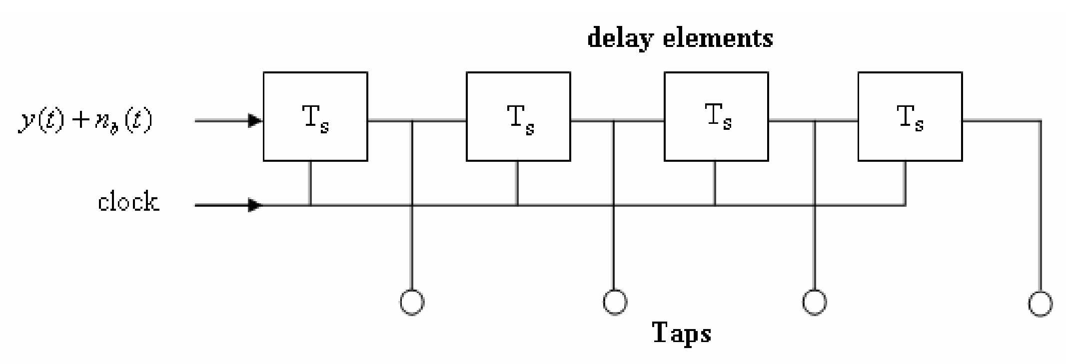


Figure 3.4 Basic linear transversal equalizer structure [1]

Assuming that the delay elements have unity gain and delay T_s, the transfer function of a linear transversal equalizer can be written as a function of the delay operator

$\exp(-j\omega T_s)$ or z^{-1}. The simplest LTE uses only feed forward taps, and the transfer function of the equalizer filter is polynomial in z^{-1}. This filter has many zeros but poles only at $z = 0$, and is called a finite impulse response (FIR) filter, or simply a transversal filter. If the equalizer has both feed forward and feedback taps, its transfer function is a rational function of z^{-1}, and is called an infinite impulse response (IIR) filter with poles and zeros. Figure 3.5 shows a tapped delay line filter with both feed forward and feedback taps. Since IIR filters tend to be unstable when used in channels where the strongest pulse arrives after an echo pulse (i.e., leading echoes), they are rarely used.

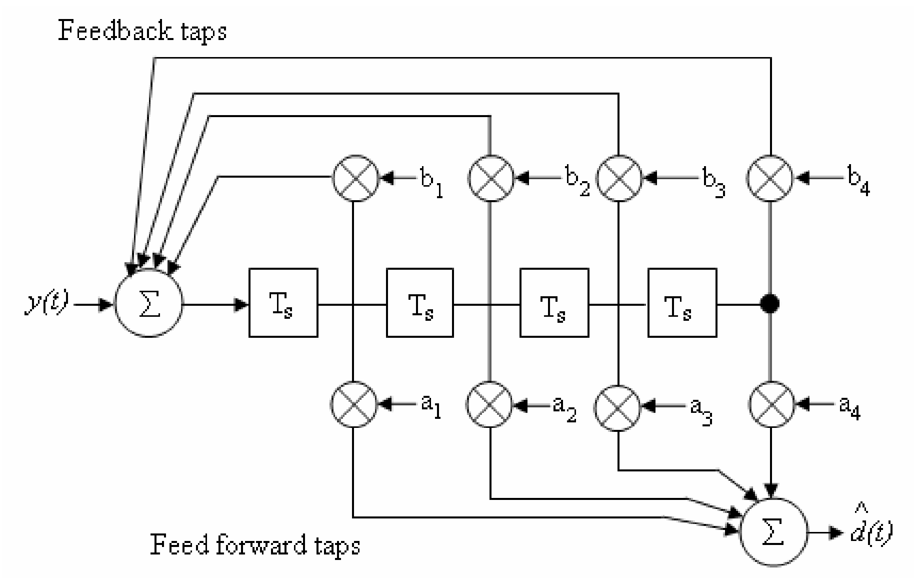


Figure 3.5 Tapped delay line filter with both feed forward and feedback taps [1]

3.5 Linear Equalizers

A linear equalizer can be implemented as an FIR filter, otherwise known as the transversal filter. This type of equalizer is the simplest type available [20]. In such an equalizer, the current and past values of the received signal are linearly weighted by the filter coefficients and summed to produce the output as shown in Figure 3.6.

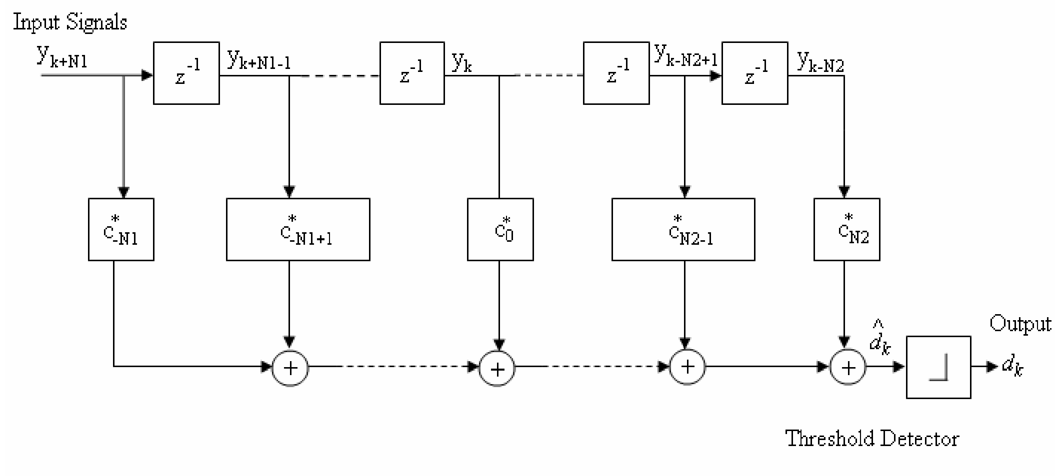


Figure 3.6 Structure of a linear transversal equalizer [1]

If the delays and the tap gains are analog, the continuous output of the equalizer is sampled at the symbol rate and the samples are applied to the decision device. The implementation is, however, usually carried out in the digital domain where the samples of the received signal are stored in a shift register. The output of this transversal filter before a decision is made is

$$\hat{d}_k = \sum_{n=-N_1}^{N_2} (c_n{*}) y_{k-n} \tag{3.20}$$

where $c_n{*}$ represents the complex filter coefficients or tap weights, $\hat{d}_k$ is the output at time index k, y_i is the input received signal at time $t_0 + iT$, t_0 is the equalizer starting time, and $N = N_1 + N_2 + 1$ is the number of taps. The values N_1 and N_2 denote the number of taps used in the forward and reverse portions of the equalizer, respectively. The minimum mean squared error $E\left[\left|e(n)\right|^2\right]$ that a linear transversal equalizer can achieve is

$$E\left[\left|e(n)\right|^2\right] = \frac{T}{2\pi} \int_{-\pi/T}^{\pi/T} \frac{N_0}{\left|F(e^{j\omega T})\right|^2 + N_0} d\omega \tag{3.21}$$

where $F(e^{j\omega T})$ is the frequency response of the channel, and N_0 is the noise power spectral density.

The linear equalizer can also be implemented as a lattice filter, whose structure is shown in Figure 3.7.

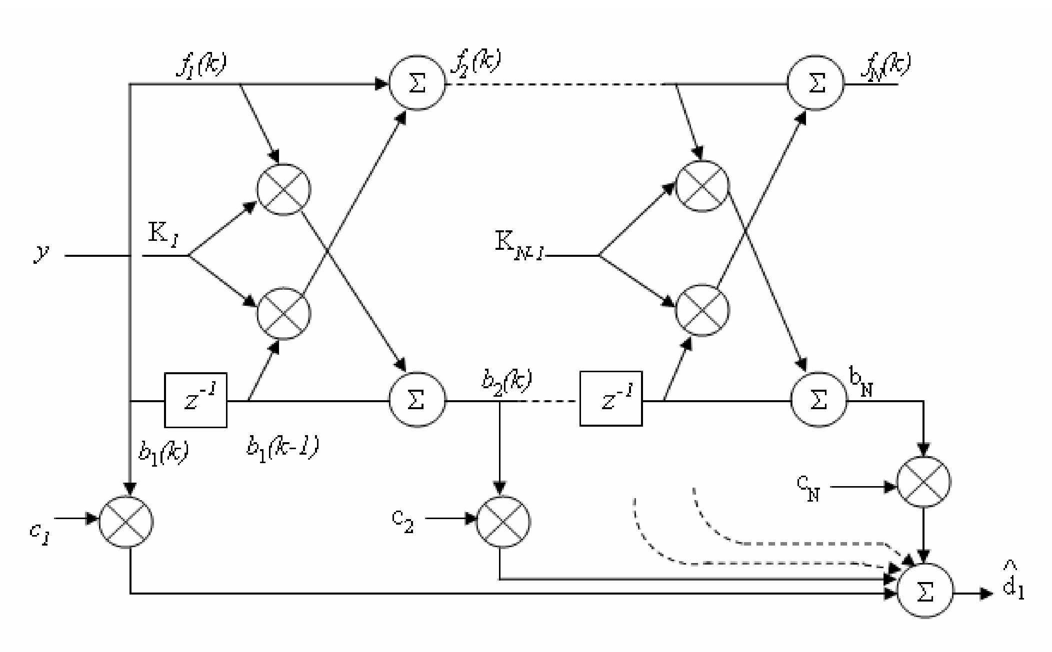


Figure 3.7 The structure of a lattice equalizer [1]

In a lattice filter, the input signal y_k is transformed into a set of N intermediate forward and backward error signals, $f_n(k)$ and $b_n(k)$ respectively, which are used as inputs to the tap multipliers and are used to calculate the updated coefficients. Each stage of the lattice is then characterized by the following recursive equations:

$$f_1(k) = b_1(k) = y(k) \tag{3.22}$$

$$f_n(k) = y(k) - \sum_{i=1}^{n} K_i y(k-i) = f_{n-1}(k) + K_{n-1}(k) b_{n-1}(k-1) \tag{3.23}$$

$$\begin{aligned} b_n(k) &= y(k-n) - \sum_{i=1}^{n} K_i y(k-n+i) \\ &= b_{n-1}(k-1) + K_{n-1}(k) f_{n-1}(k) \end{aligned} \tag{3.24}$$

where $K_n(k)$ is the reflection coefficient for the nth stage of the lattice. The backward error signals, b_n, are then used as inputs to the tap weights, and the output of the equalizer is given by

$$\hat{d}_k = \sum_{n=1}^{N} c_n(k) b_n(k) \tag{3.25}$$

Two main advantages of the lattice equalizer are their numerical stability and faster convergence. Also, the unique structure of the lattice filter allows the dynamic assignment of the most effective length of the lattice equalizer. Hence, if the channel is not very time dispersive, only a fraction of the stages are used. When the channel becomes more time dispersive, the length of the equalizer can be increased by the algorithm without stopping the operation of the equalizer [22]. The structure of a lattice equalizer, however, is more complicated than a linear transversal equalizer.

3.6 Nonlinear Equalizers

Nonlinear equalizers are used in applications where the channel distortion is too severe for a linear equalizer to handle, and are commonplace in practical wireless systems. Linear equalizers do not perform well on channels which have deep spectral nulls in the passband. In an attempt to compensate for the distortion, the linear equalizer places too much gain in the vicinity of the spectral null, thereby enhancing the noise present in those frequencies.

Three very effective nonlinear methods have been developed which offer improvements over linear equalization techniques and are used in most 2G and 3G systems. These are:

1. Decision Feedback Equalization (DFE)
2. Maximum Likelihood Symbol Detection
3. Maximum Likelihood Sequence Estimation (MLSE)

3.6.1 Decision Feedback Equalization

The basic idea behind decision feedback equalization is that once an information symbol has been detected and decided upon, the ISI that it induces on future symbols can be estimated and subtracted out before detection of subsequent symbols. The DFE can be realized in either the direct transversal form or as a lattice filter. The direct form is shown in Figure 3.8.

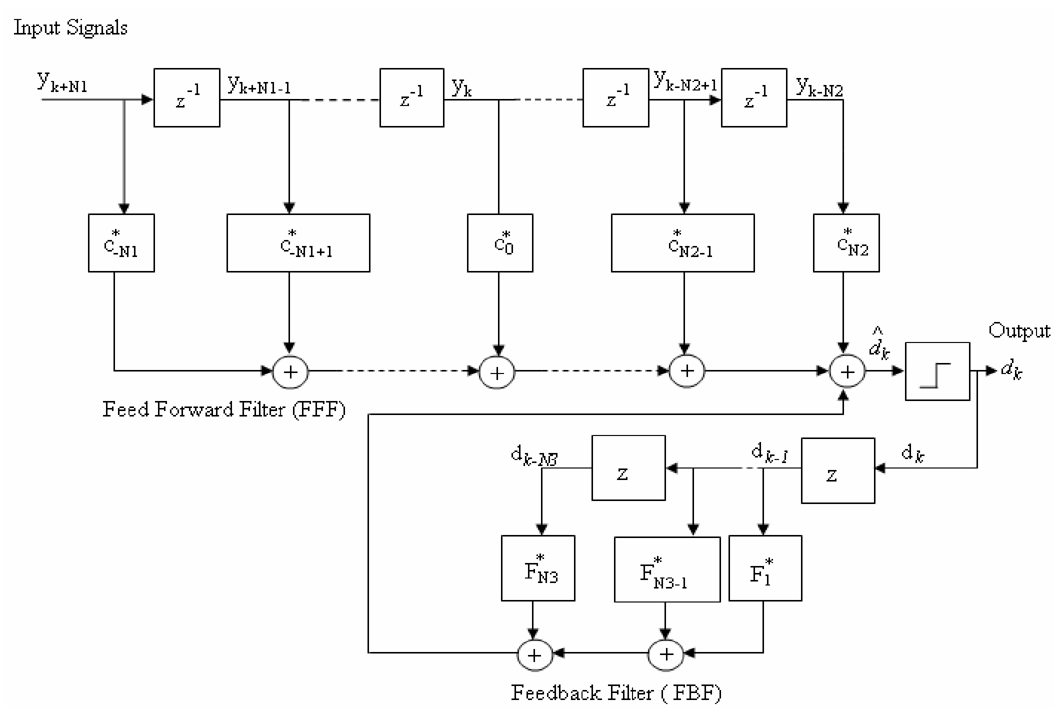


Figure 3.8 Decision feedback equalizer (DFE) [1]

It consists of a feed forward filter (FFF) and a feedback filter (FBF). The FBF is driven by decisions on the output of the detector, and its coefficients can be adjusted to cancel the ISI on the current symbol from past detected symbols. The equalizer has $N_1 + N_2 + 1$ taps in the feed forward filter and N_3 taps in the feedback filter, and its output can be expressed as:

$$\hat{d}_k = \sum_{n=-N_1}^{N_2} c_n^* y_{k-n} + \sum_{i=1}^{N_3} F_i d_{k-i} \tag{3.26}$$

where c_n^* and y_n are tap gains and the inputs, respectively, to the feed forward filter, F_i^* are tap gains for the feedback filter, and $d_i(i < k)$ is the previous decision made on the detected signal. That is, once $\hat{d}_k$ is obtained using Equation (3.26), d_k along with previous decisions d_{k-1}, d_{k-2}, … are fed back into the equalizer [1], and $\hat{d}_{k+1}$ is obtained using Equation (3.26).

The minimum mean square error a DFE can achieve is

$$E\left[\left|e(n)\right|^2\right]_{\min} = \exp\left\{\frac{T}{2\pi}\int_{-\pi/T}^{\pi/T} \ln\left[\frac{N_0}{\left|F(e^{j\omega T})\right|^2 + N_0}\right] d\omega\right\} \tag{3.27}$$

It can be shown that the minimum MSE for a DFE in Equation (3.27) is always smaller that that of an LTE in Equation (3.21) unless $|F(e^{j\omega T})|$ is a constant (i.e., when adaptive equalization is not needed). If there are nulls in $|F(e^{j\omega T})|$, a DFE has significantly smaller minimum MSE than an LTE. Therefore, an LTE is well behaved when the channel spectrum is comparatively flat, but if the channel is severely distorted or exhibits nulls in the spectrum, the performance of an LTE deteriorates and the mean squared error of a DFE is much better than an LTE. Also, an LTE has difficulty equalizing a nonminimum phase channel, where the strongest energy arrives after the first arriving signal component [1]. Thus, a DFE is more appropriate for severely distorted wireless channels.

The lattice implementation of the DFE is equivalent to a transversal DFE having a feed forward filter of length N_1 and a feedback filter of length N_2, where $N_1 > N_2$.

Another form of DFE proposed by Belfiore and Park is called a predictive DFE and is shown in Figure 3.9.

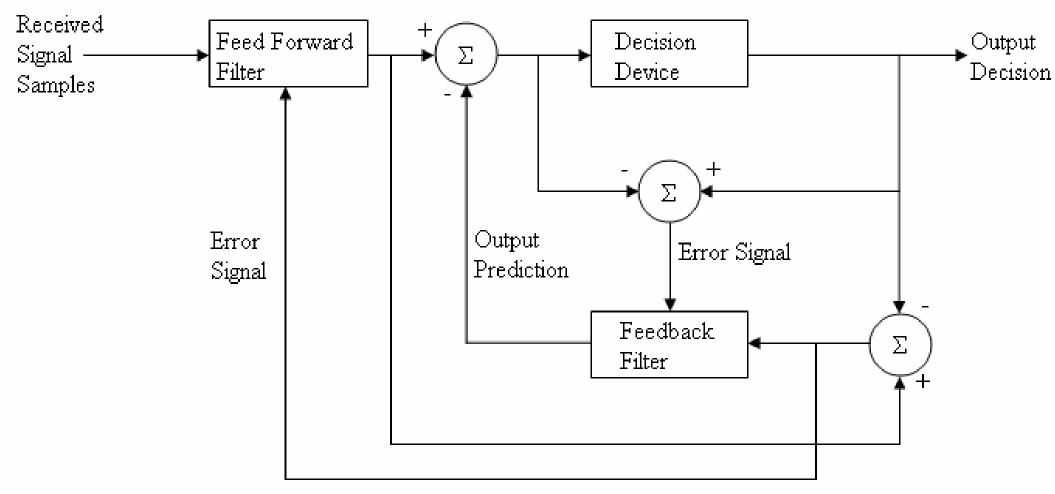


Figure 3.9 Predictive decision feedback equalizer [1]

It also consists of a feed forward filter (FFF) as in the conventional DFE. However, the feedback filter (FBF) is driven by an input sequence formed by the difference of the output of the detector and the output of the feed forward filter. Hence, the FBF here is called a noise predictor because it predicts the noise and the residual ISI contained in the signal at the FFF output and subtracts from it the detector output after some feedback delay. The predictive DFE performs as well as the conventional DFE as the limit in the number of taps in the FFF and the FBF approach infinity. The FBF in the predictive DFE can also be realized as a lattice structure. The RLS lattice structure (discussed in Section 3.7.4) can be used in this case to yield fast convergence.

3.6.2 Maximum Likelihood Sequence Estimation Equalizer

The MSE-based linear equalizers described previously are optimum with respect to the criterion by minimum probability of symbol error when the channel does not introduce any amplitude distortion. Yet this is precisely the condition in which an equalizer is needed for a mobile communications link. This limitation on MSE-based equalizers led researchers to investigate optimum or nearly optimum nonlinear structures. These equalizers use various forms of the classical maximum likelihood receiver structure. Using a channel impulse response simulator within the algorithm, the MLSE tests all possible data sequences (rather than decoding each received symbol by itself), and chooses the data sequence with the maximum probability as the output. An MLSE usually has a large computational requirement, especially when the delay spread of the channel is large. Using the MLSE as an equalizer was first proposed by Forney [23] in which he set up a basic MLSE estimator structure and implemented it with the Viterbi algorithm. This algorithm was recognized to be a maximum likelihood sequence estimator (MLSE) of the state sequences of a finite state Markov process observed in memoryless noise. It has recently been implemented successfully for equalizers in mobile radio channels.

The MLSE can be viewed as a problem in estimating the state of a discrete-time finite state machine, which in this case happens to be the radio channel coefficients f_k, and with a channel state which at any instant of time is estimated by the receiver based on the L most recent input samples. Thus, the channel has M^L states, where M is the size of the symbol alphabet of the modulation. That is, an M^L trellis is used by the receiver to model the channel over time. The Viterbi algorithm then tracks the state of the channel by the paths through the trellis and gives at stage k a rank ordering of the M^L most probable sequences terminating in the most recent L symbols.

The block diagram of a MLSE receiver based on the DFE is shown in Figure 3.10.

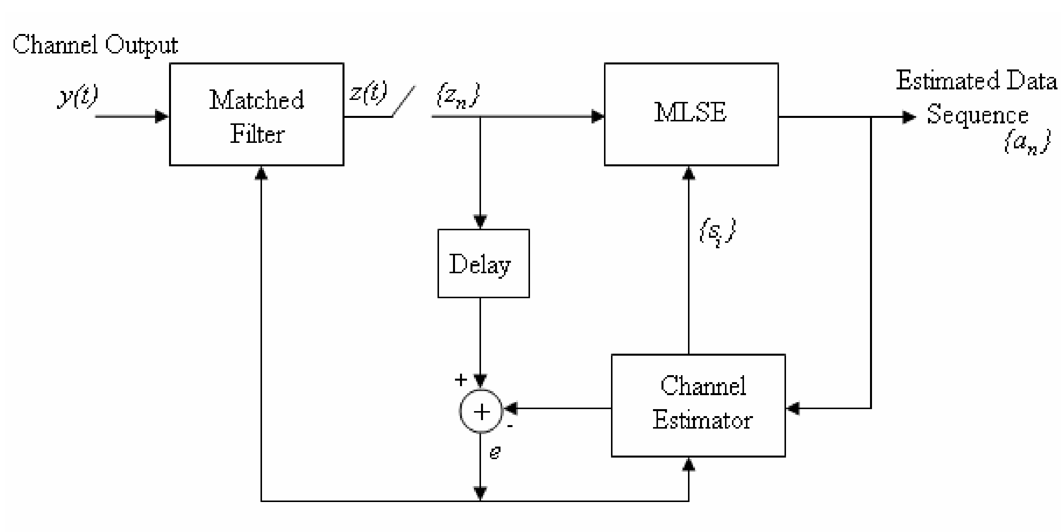


Figure 3.10 The structure of a maximum likelihood sequence estimation (MLSE) equalizer with an adaptive matched filter [1]

The MLSE is optimal in the sense that it minimizes the probability of a sequence error. The MLSE requires knowledge of the channel characteristics in order to compute the metrics for making decisions. The MLSE also requires knowledge of the statistical distribution of the noise corrupting the signal. Thus, the probability distribution of the noise determines the form of the metric for optimum demodulation of the received signal. It should be noted that the matched filter operates on the continuous time signal, whereas the MLSE and channel estimator rely on discretized (nonlinear) samples.

3.7 Adaptive Equalizer Algorithms

Since an adaptive equalizer compensates for an unknown and time-varying channel, it requires a specific algorithm to update the equalizer coefficients and track channel variations. A wide range of algorithms exists to adapt the filter coefficients. This section describes some practical issues regarding equalizer algorithm design, and outlines three of the basic algorithms for adaptive equalization. The performance of an algorithm is determined by various factors, such as:

- **Rate of convergence** – This is the number of iterations required for the algorithm, in response to fixed inputs, to converge close enough to the optimum solution. A fast rate of convergence allows the algorithm to adapt quickly to a fixed environment of unknown statistics. Furthermore, it enables the algorithm to track statistical variations when operating in a non-stationary environment.

- **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.

- **Computational complexity** – This is the number of operations necessary to make one complete iteration of the algorithm.

- **Numerical properties** – When an algorithm is implemented numerically, inaccuracies are produced due to round-off noise and representation errors in the computer. These kinds of errors influence the stability of the algorithm.

3.7.1 Factors Affecting Equalizer Choice and Its Algorithm

In reality, the cost of the computing platform, the power budget, and the radio propagation characteristics have a great influence in deciding the choice of an equalizer and its algorithm. In portable radio applications (i.e., cellular mobile phones), current consumption (battery drain) is a vital consideration to maximize talk-time. Therefore, equalizers are implemented if and only if they can provide enough link improvement to justify the cost and power burden. The radio channel characteristics and intended use of the subscriber equipment are also key factors. The speed of the mobile unit determines the channel fading rate and the Doppler shift, which is directly related to the coherence time of the channel. The choice of algorithm and its corresponding rate of convergence depend on the channel data rate and coherence time.

3.7.2 Zero Forcing Algorithm

In a zero forcing equalizer, the equalizer coefficients c_n are chosen to force the samples of the combined channel and equalizer impulse response to zero at all but one of the NT spaced sample points in the tapped delay line filter. By letting the number of coefficients increase without bound, an infinite length equalizer with zero ISI at the output can be obtained. Where each of the delay elements provides a time delay equal to the symbol duration T, the frequency response $H_{eq}(f)$ of the equalizer is periodic with a period equal to the symbol rate $1/T$. The combined response of the channel with the equalizer must satisfy Nyquist's first criterion

$$H_{ch}(f)H_{eq}(f) = 1, \quad |\mathrm{f}| < 1/2T \tag{3.28}$$

where $H_{ch}(f)$ is the folded frequency response of the channel. Thus, an infinite length, zero ISI equalizer is simply an inverse filter, which inverts the folded frequency response of the channel. This infinite length equalizer is usually implemented by a truncated length version.

The zero forcing algorithm was developed by Lucky [24] for wired communications. The zero forcing equalizer has the disadvantage that the inverse filter may excessively amplify noise at frequencies where the folded channel spectrum has high attenuation. The ZF equalizer, therefore, ignores the effect of noise, and is not usually used for wireless links. However, it performs well for static channels with high SNR, such as PSTN phone lines [6].

3.7.3 Least Mean Square Algorithm

A more robust equalizer is the LMS equalizer. The goal of this type of equalizer is to minimize the mean square error (MSE) between the desired and the actual equalizer outputs. From Figure 3.2, the prediction error can be written as:

$$e_k = d_k - \hat{d}_k = x_k - \hat{d}_k \tag{3.29}$$

and using Equation (3.10)

$$e_k = x_k - y_k^T w_k = x_k - w_k^T y_k \tag{3.30}$$

Equation (3.12) is squared in order to compute the mean square error $|e_k|^2$ at time instant k to yield

$$\zeta = E[e_k^* e_k] \tag{3.31}$$

The LMS algorithm attempts to minimize the mean square error defined in Equation 3.31. For a specific channel condition, the prediction error e_k is dependent on the tap gain vector w_N, so the MSE of an equalizer is a function of w_N. The cost function $J(w_N)$ denotes the mean squared error as a function of tap gain vector w_N. To minimize the MSE, the derivative of Equation (3.32) needs to be set to zero.

$$\frac{\partial}{\partial w_N} J(w_N) = -2 p_N + 2 R_{NN} w_N = 0 \tag{3.32}$$

Simplifying Equation (3.32) yields

$$R_{NN} \hat{w}_N = p_N \tag{3.33}$$

Equation (3.33) is called the normal equation since the error is minimized and made normal (orthogonal) to the projection related to the desired signal x_k. When Equation (3.33) is satisfied, the minimum MSE of the equalizer becomes

$$J_{opt} = J(\hat{w}_N) = E[x_k x_k^{*}] - p_N{}^{T} \hat{w}_N \tag{3.34}$$

There exist many variants of the LMS algorithm by solving Equation (3.34). The easiest and the most straightforward one is to calculate

$$\hat{w} = R_{NN}{}^{-1} p_N \tag{3.35}$$

In practice, the minimization of the MSE is carried out recursively, and may be performed by use of the gradient descent algorithm. This is more commonly called the least mean square (LMS) algorithm. The LMS algorithm is the simplest equalization algorithm and requires only $2N+1$ operations per iteration [2].

3.7.4 Recursive Least Squares Algorithm

The convergence rate of the gradient-based LMS algorithm is slow. Faster converging algorithms are based on a least squares approach, as opposed to the statistical approach used in the LMS algorithm. That is, rapid convergence relies on error measures expressed in terms of a time average of the actual received signal instead of a statistical average. This leads to the family of powerful, albeit complex, adaptive signal processing methods known as recursive least squares (RLS), which significantly improves the convergence of adaptive equalizers.

The RLS algorithm may be summarized as follows:

1. Initialize w (0) = k (0) = x (0) = 0, $R^{-1}(0) = \delta I_{NN}$ where I_{NN} is an N×N identity matrix, and δ is a large positive constant.
2. Recursively compute the following:

$$\hat{d}(n) = \omega_N^{\ T}(n-1)y(n) \tag{3.36}$$

$$e(n) = x(n-1) - \hat{d}(n) \tag{3.37}$$

$$k(n) = \frac{R^{-1}(n-1)y(n)}{\lambda + y^T(n)R^{-1}(n-1)y(n)} \tag{3.38}$$

$$R^{-1}(n) = \frac{1}{\lambda}\left[R^{-1}(n-1) - k(n)y^T(n)R^{-1}(n-1)\right] \tag{3.39}$$

$$\omega(n) = \omega(n-1) + k_N(n)e^*(n) \tag{3.40}$$

In Equation (3.38), λ is the weighting coefficient that can change the performance of the equalizer. If a channel is time-invariant, λ can be set to one. Usually $0.8 < \lambda < 1$ is used. The value of λ has no influence on the rate of convergence, but does determine the tracking ability of the RLS equalizers. The smaller the λ, the better the tracking ability of the equalizer. However, if λ is too small, the equalizer will be unstable. The RLS algorithm described above, called the Kalman RLS algorithm, uses $2.5N^2 + 4.5N$ arithmetic operations per iteration.

3.7.6 Viterbi Algorithm

The Viterbi algorithm was first proposed by Andrew J. Viterbi in 1967 as a solution to the decoding of convolutional codes [23]. The Viterbi algorithm is a fundamental signal processing technique widely used in modern digital communication systems. Typical application examples are maximum likelihood sequence estimation (MLSE) in the presence of intersymbol interference (ISI) and decoding of convolutional codes. The Viterbi algorithm can be described as an algorithm that finds the most likely path through a trellis diagram, given a set of observations. The trellis diagram is a representation of a finite set of states from a Finite States Machine. Each node in the diagram represents a state and each edge a possible transition between two states at consecutive discrete time intervals. Viterbi decoding is an optimal solution for detecting sequences from a channel with intersymbol interference (ISI). But if the interference is severe (i.e. the time spread is high), it does not perform well enough. Therefore the need for equalizing the signal before attempting to decode it arises. As the channel is typically time variant, the equalization needs to be adaptive. A first choice can be to provide the equalizer with its own decision device, and feed these decisions back to the equalizer for tap coefficient updating, making equalizer and Viterbi detection independent of each other. However, the predecoding decisions are likely to be fairly unreliable, due to the low signal-to-noise ratio; then employing the Viterbi decoder's decisions to update the equalizer's tap weights can be considered. Viterbi decisions are more reliable and will then provide an appropriate feedback to the adaptive equalizer, if the latter is linear. To get around with the delay introduced by the Viterbi decoder, an analogous delay can be introduced before creating the error signal.

3.7.6 Comparative Analysis of Algorithms

In this last section of chapter 3, after explaining different equalizer structures in detail, comparative analysis of different algorithms is given. These algorithms, their corresponding number of multiplication operations per iteration, advantages and disadvantages are summarized in Table 3.1.

Table 3.1 Comparison chart for different adaptive equalizer algorithms [1]

Algorithm	Number of Multiplication Operations	Advantages	Disadvantages
LMS Gradient DFE	$2N+1$	Low computational complexity, simple program	Slow convergence, poor tracking
Kalman RLS	$2.5N^2+4.5N$	Fast convergence, good tracking ability	High computational complexity
FTF	$7N+14$	Fast convergence, good tracking, low computational complexity	Complex programming, unstable
Gradient Lattice	$13N-8$	Stable, low computational complexity, flexible structure	Performance not as good as other RLS, complex programming
Gradient Lattice DFE	$13N_1+33N_2-36$	Low computational complexity	Complex programming
Fast Kalman DFE	$20N+5$	Can be used for DFE, fast convergence and good tracking	Complex programming, computation not low, unstable
Square Root RLS DFE	$1.5N^2+6.5N$	Better numerical properties	High computational complexity

3.8 Summary

In this chapter, equalization basics, different equalization techniques as well as algorithms were presented. There are many variations of the LMS and RLS algorithms in use for adapting an equalizer. Table 3.1 outlines the computational complexities of different algorithms, and lists some advantages and disadvantages of each algorithm. The RLS algorithms have similar convergence and tracking performances, which are much better than the LMS algorithm. However, these RLS algorithms usually have high computational requirements and demand complex programming. Additionally, some RLS algorithms tend to be unstable.

4. DESIGN OF FREQUENCY DOMAIN ADAPTIVE EQUALIZER

4.1 Overview

A frequency-domain adaptive equalizer is basically composed of a FIR adaptive filter whose coefficients are adjusted adaptively to achieve the inverse transfer function of the fading channel as close as possible (Refer to Equation 3.5). This process is accomplished by using a training sequence as explained in detail in chapter 3. The basic operation underlying a frequency-domain adaptive filter is the transformation of the input signal into a more desirable form before the adaptive processing. This is accomplished by one or more discrete Fourier transforms (DFTs) or filter banks whereby the input signal is transformed to the frequency domain. In this chapter, details of the design stages of both the multipath Rayleigh fading channel and the frequency domain adaptive equalizer will be explained.

The contribution of this thesis is that the LMS algorithm has been applied to adaptive equalization in the frequency domain for the multipath Rayleigh fading channel. Previous work considered using the LMS algorithm for adaptive equalization in the time domain. Current GSM receivers implement the MLSE equalizer with Viterbi algorithm in the time domain [1].

4.2 Equalization in the Frequency Domain

Filtering can be done directly in the frequency domain, by operating on the signal's frequency spectrum. Figure 4.1 shows how a noisy sine wave can be cleaned up by operating directly upon its frequency spectrum to select only a range of frequencies that include signal frequency components but exclude much of the noise:

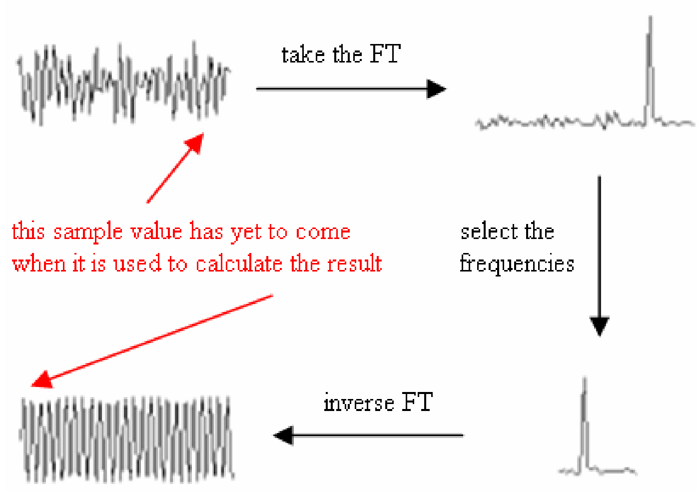


Figure 4.1 Equalization in the frequency domain by delaying

- The noisy sine wave (shown as a time signal) contains narrow band signal plus broad band noise.
- The frequency spectrum is calculated.
- The frequency spectrum is modified by suppressing a range outside the signal's frequency components.
- The time domain signal is calculated from the frequency spectrum.
- The resulting signal (shown in the time domain again) looks much cleaner.

Filtering in the frequency domain is efficient because every calculated sample of the filtered signal takes account of all the input samples. Because the frequency spectrum contains information about the whole of the signal - for all time values - samples early in the output take account of input values that are late in the signal, and so can be thought of as still to happen. The frequency domain filter looks ahead to see what the signal is going to do. In the design process of the adaptive equalizer, the input samples are delayed by 16 samples before starting the filter calculation.

4.3 Preamble Training Sequence

The word ‘preamble’ means the introductory, the initial portion of a bigger part. In our design, the first 444 dummy bits (which both the GSM handset and the base station have knowledge about the undistorted version) of the 2072 random bits are used for training the equalizer. The adaptive equalizer uses both the distorted and the undistorted training patterns with LMS Gradient algorithm to minimize the MSE between the equalizer output and the desired training signal. Once the training is completed, the remaining 1628 data bits that carry the message signal are equalized to their original states (ISI free) as close as possible.

4.4 Application of FDAE in GSM Mobile Channel

In this thesis, a frequency domain adaptive equalizer is proposed and implemented to be used in a GSM mobile channel. To do this, it is necessary to simulate the Rayleigh multipath fading channel (create ISI) and then use the FDAE to equalize the distorted signal. Sections 4.4.1 and 4.4.2 outline the details of the implementation of these two components of the design process.

4.4.1 Multipath Rayleigh Fading Channel Model for GSM

The multipath Rayleigh fading channel for GSM is created using the MATLAB built-in function `rayleighchan().rayleighchan()` function accepts and requires four different parameters to create the channel object `chan`.

```
chan = rayleighchan(ts, fd, delay, gain)
```

These four parameters are the symbol time, Doppler frequency, and path delay and gain vectors. The symbol time is calculated from the symbol rate of GSM which is 270.833 kb/s. Thus a symbol time of approximately 3.69 µs is used. Doppler frequency is chosen to be 4 Hz. This value is chosen based on statistical data [14] and reflects a mobile user moving at the walking speed. Table 4.1 shows the path delay and gain values used for this channel.

Table 4.1 Multipath rayleigh fading channel path delays and gains

Path Number	Path Delay (μs)	Path Gain (dB)
1	0.1	-0.9
2	0.2	-1.7
3	0.3	-2.6
4	0.4	-3.5
5	0.5	-4.3
6	0.6	-5.2
7	0.7	-7.8
8	0.8	-10.2
9	0.9	-18.0
10	1.7	-22.4

4.4.2 FDAE System for GSM Receiver

Initially, the first 444 (preamble) bits of 2072 random bits used for training the adaptive filter were generated using the random number generator (RG) in MATLAB. Then the signal was modulated using the GSM modulation scheme, Gaussian Minimum Shift Keying. Next, the modulated signal was passed through the Rayleigh Multipath fading channel to introduce channel distortion and create ISI. At the GSM receiver end, the received signal is then demodulated to obtain the distorted 2072 random bits. The adaptive filter has the original undistorted 444-bit training sequence stored in it. Using LMS algorithm, the equalizer adaptively processes the distorted initial 444 bits of the demodulated signal and updates its coefficients until the filter output matches the original training sequence as close as possible (Refer to Equation 3.5). Once the MSE decreases to approximately zero or the predefined value, filter coefficients are stored by the system. At this point, the frequency domain adaptive filter has knowledge about the characteristics of the multipath Rayleigh fading channel. Now, the equalizer has the ability to equalize the remaining of the data bits to their original states. Figure 4.2 shows the complete structure of the frequency domain adaptive equalizer implemented for the GSM receiver. The adaptive filter section of the equalizer is shown in Figure 4.3.

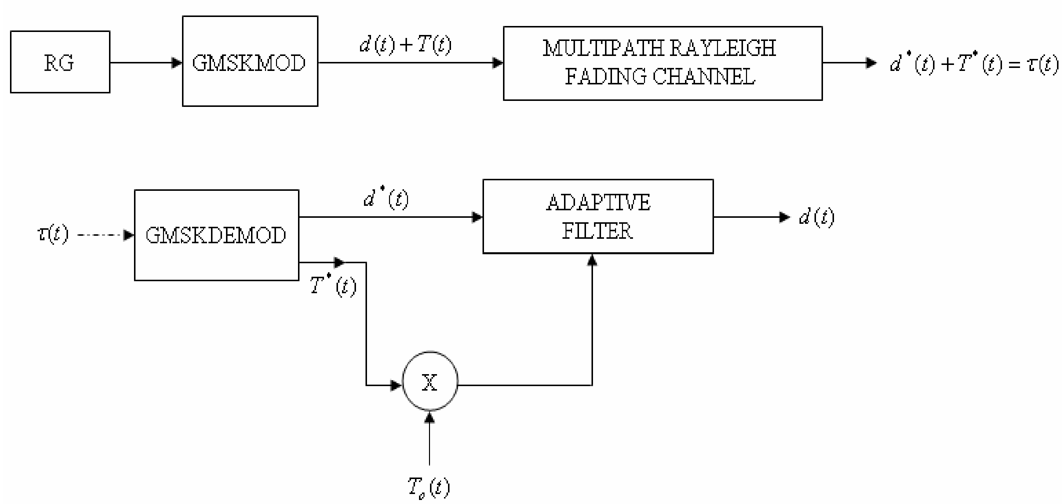


Figure 4.2 Frequency domain adaptive equalizer system for GSM receiver

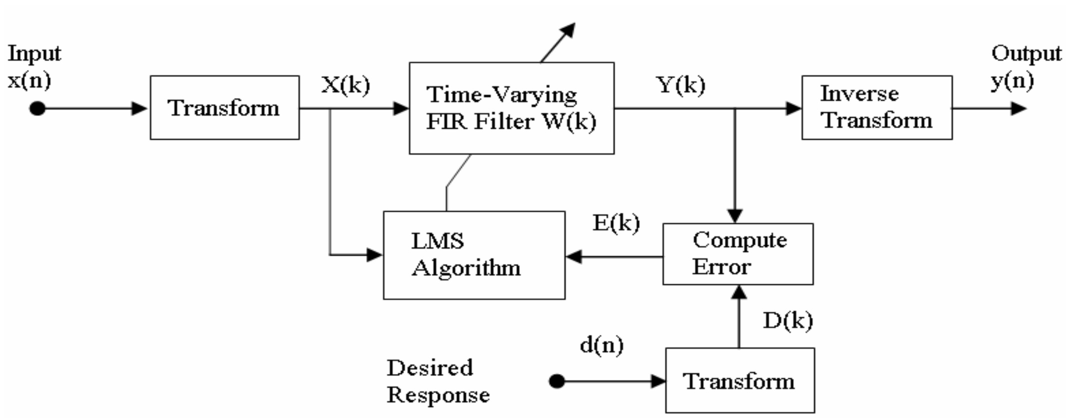


Figure 4.3 Frequency domain adaptive filter structure

$$W(k) = W_0(k),...,W_{N-1}(k) \quad (4.1)$$

$$X(k) = X_0(k),...,X_{N-1}(k) \quad (4.2)$$

$$Y(k) = X(k)W(k) \quad (4.3)$$

$$W(k+1) = W(k) + 2\mu X(k)E(k) \quad (4.4)$$

$X(k)$, $W(k)$, $Y(k)$ and $D(k)$ are the input signal, filter coefficient weight, filter output, and desired signal vectors in the frequency domain respectively. The frequency domain adaptive filter structure shown above is implemented in MATLAB using the `adaptfilt.fdaf()` function. `adaptfilt.fdaf()` requires and accepts four parameters in order to construct the frequency domain adaptive filter (FDAF) `ha`:

```
ha = adaptfilt.fdaf(32,mu,1,del,lam)
```

These parameters are the number of taps (N=32), initial FFT input powers (del=1), averaging factor (lam=0.9) and the LMS algorithm step size (μ=0.1). At the end of training the filter, the FDAF returns the filter coefficients necessary to reverse the effect of the multipath Rayleigh fading channel.

4.5 Summary

In this chapter, the main sections of the adaptive equalizer as well as the multipath Rayleigh fading channel model implemented were explained. Since FDAE uses the LMS Gradient algorithm for adaptation, it is numerically efficient and requires only 2N + 1 multiplication operations per iteration. In chapter 5, the performance graphs of the implemented equalizer using four different test profiles as well as the BER comparison table before and after the equalization process will be presented.

5. RESULTS AND DISCUSSIONS

5.1 Overview

Following the modeling of the Rayleigh multipath channel and implementing the frequency domain adaptive equalizer in MATLAB as explained in chapter 4, four different multipath Rayleigh fading channel models are used to test the equalizer performance; Test Case I, II, III, IV uses 3, 5, 7, 10 path Rayleigh fading channel models, respectively. In this chapter, four different simulation results will be shown. Each figure in Sections 5.2 to 5.5 shows the received signal, the desired signal, equalized signal and the noise (error) signal.

5.2 Three-path Rayleigh Fading Channel Simulation Results

In this simulation, a three-path Rayleigh fading channel is implemented using a Doppler frequency of 4 Hz. Path delay and gain values are given in Table 5.1.

Table 5.1 Path delay and gain values for the three-path rayleigh fading channel model

Path Number	Path Delay (µs)	Path Gain (dB)
1	0.1	-0.9
2	0.2	-1.7
3	0.3	-2.6

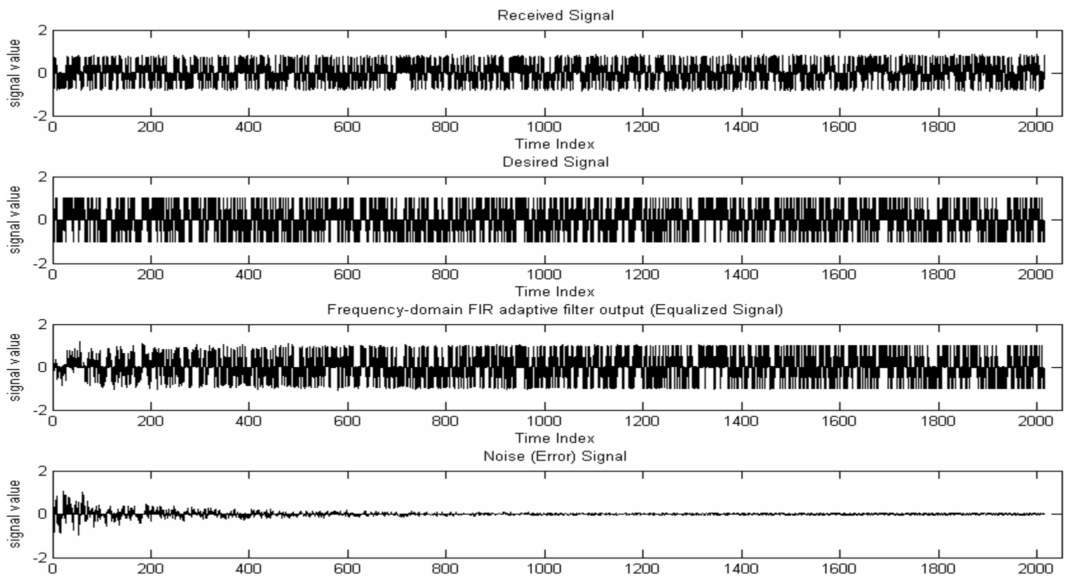
The following command is executed in MATLAB to yield the performance of the equalizer:

```
[number_noeq, ratio_noeq, number_eq, ratio_eq] = multipathadapt(3)
```

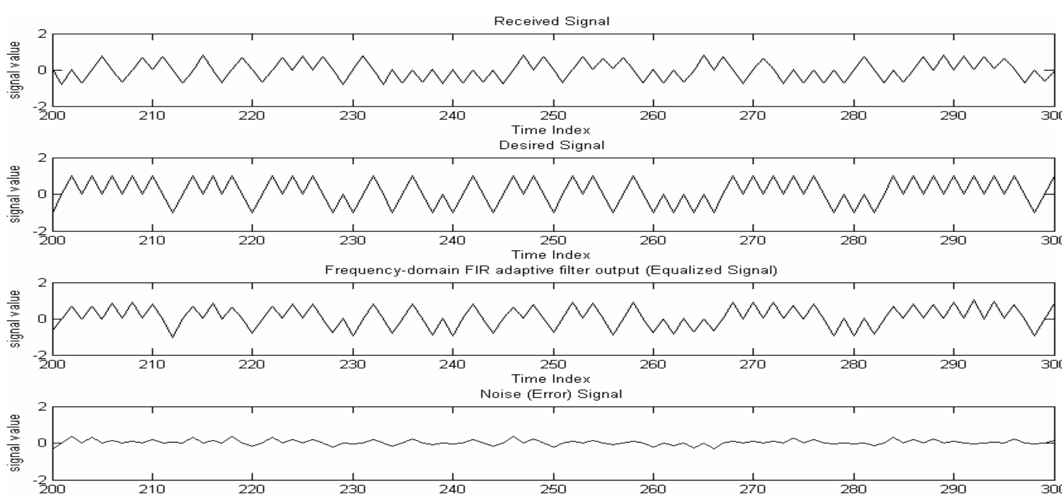
The MATLAB script `multipathadapt()` returned the following results. The performance graph is shown in Figure 5.1.

number_noeq	= 1037	(# of bits interpreted incorrectly without equalization)
ratio_noeq	= 0.5063	(BER without equalization)
number_eq	= 38	(# of bits interpreted incorrectly with equalization)
ratio_eq	= 0.0186	(BER with equalization)

Elapsed time is 0.00039543 seconds.



(a)



(b)

Figure 5.1 Equalizer performance graph for a three-path Rayleigh fading channel (a) Samples from 0 to 2072 (b) Samples from 200 to 300

5.3 Five-path Rayleigh Fading Channel Simulation Results

In this simulation, a five-path Rayleigh fading channel is implemented using a Doppler frequency of 4 Hz. Path delay and gain values are given in Table 5.2.

Table 5.2 Path delay and gain values for the five-path rayleigh fading channel model

Path Number	Path Delay (μs)	Path Gain (dB)
1	0.1	-0.9
2	0.2	-1.7
3	0.3	-2.6
4	0.4	-3.5
5	0.5	-4.3

The following command is executed in MATLAB to yield the performance of the equalizer:

```
[number_noeq, ratio_noeq, number_eq, ratio_eq] = multipathadapt(5)
```

The MATLAB script `multipathadapt()` returned the following results. The performance graph is shown in Figure 5.2.

number_noeq = 997 (# of bits interpreted incorrectly without equalization)
ratio_noeq = 0.4868 (BER without equalization)
number_eq = 40 (# of bits interpreted incorrectly with equalization)
ratio_eq = 0.0195 (BER with equalization)
Elapsed time is 0.00038990 seconds.

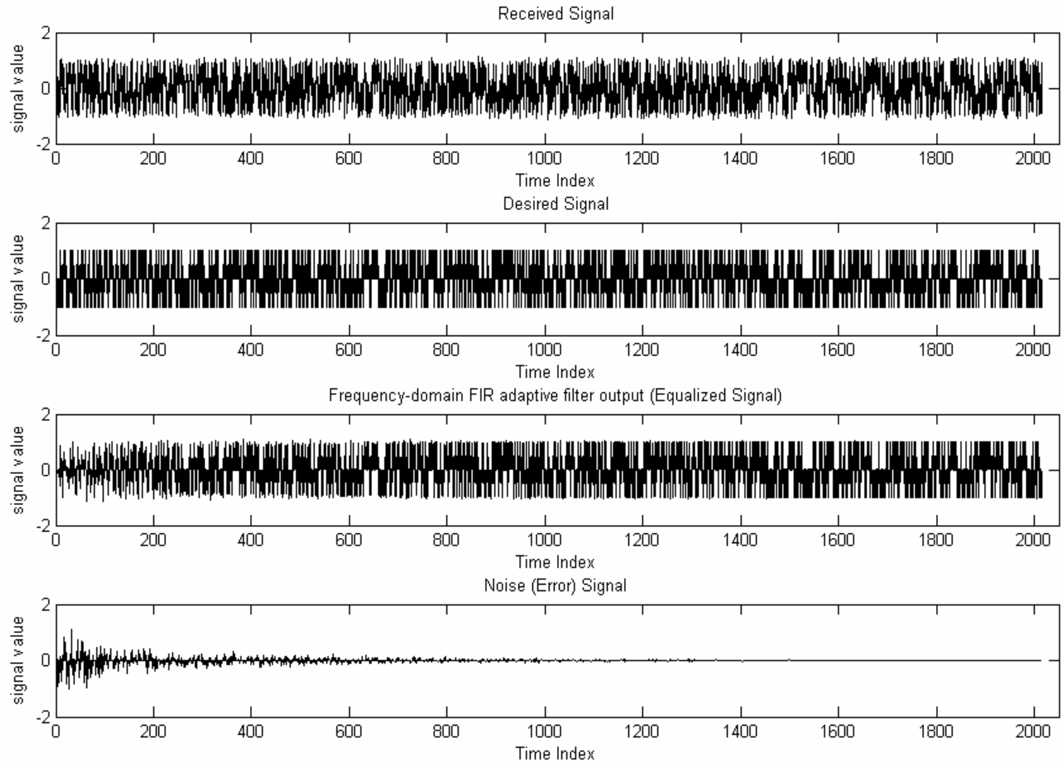


Figure 5.2 Equalizer performance graph for a five-path rayleigh fading channel

5.4 Seven-path Rayleigh Fading Channel Simulation Results

In this simulation, a seven-path Rayleigh fading channel is implemented using a Doppler frequency of 4 Hz. Path delay and gain values are given in Table 5.3.

Table 5.3 Path delay and gain values for the seven-path rayleigh fading channel model

Path Number	Path Delay (μs)	Path Gain (dB)
1	0.1	-0.9
2	0.2	-1.7
3	0.3	-2.6
4	0.4	-3.5
5	0.5	-4.3
6	0.6	-5.2
7	0.7	-7.8

The following command is executed in MATLAB to yield the performance of the equalizer:

```
[number_noeq, ratio_noeq, number_eq, ratio_eq] = multipathadapt(7)
```

The MATLAB script `multipathadapt()` returned the following results. The performance graph is shown in Figure 5.3.

number_noeq	= 1006	(# of bits interpreted incorrectly without equalization)
ratio_noeq	= 0.4912	(BER without equalization)
number_eq	= 37	(# of bits interpreted incorrectly with equalization)
ratio_eq	= 0.0181	(BER with equalization)

Elapsed time is 0.00041397 seconds.

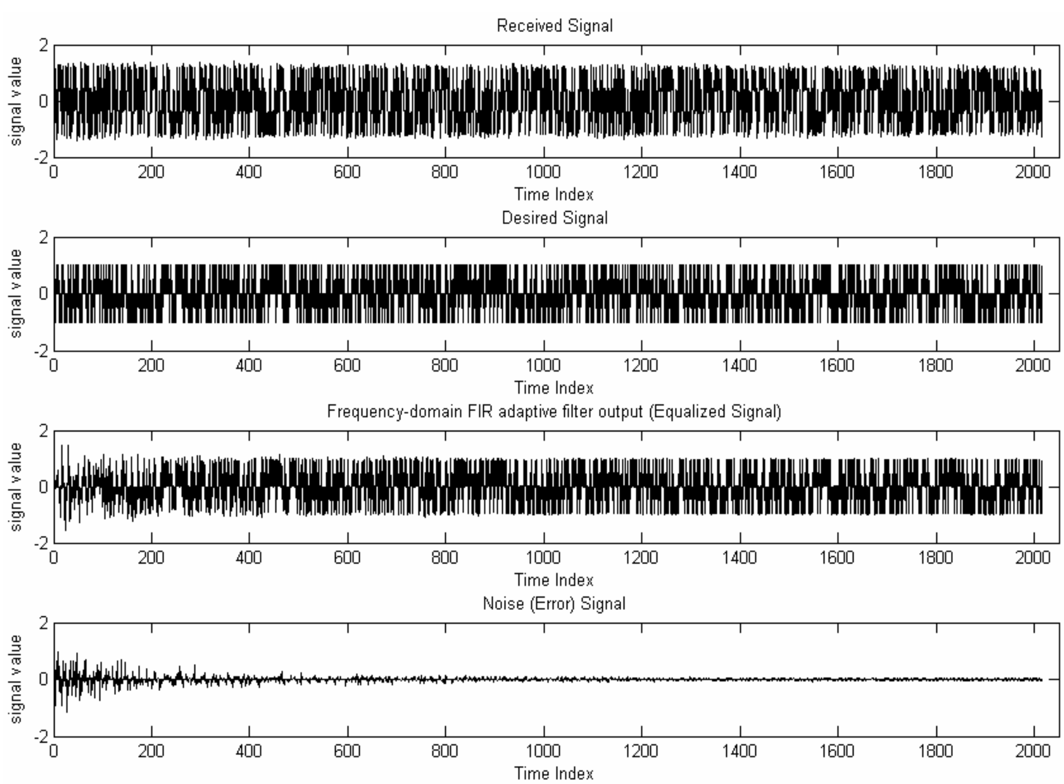


Figure 5.3 Equalizer performance graph for a seven-path rayleigh fading channel

5.5 Ten-path Rayleigh Fading Channel Simulation Results

In this simulation, a ten-path Rayleigh fading channel is implemented using a Doppler frequency of 4 Hz. Path delay and gain values are given in Table 5.4.

Table 5.4 Path delay and gain values for the ten-path rayleigh fading channel model

Path Number	Path Delay (μs)	Path Gain (dB)
1	0.1	-0.9
2	0.2	-1.7
3	0.3	-2.6
4	0.4	-3.5
5	0.5	-4.3
6	0.6	-5.2
7	0.7	-7.8
8	0.8	-10.2
9	0.9	-18.0
10	1.7	-22.4

The following command is executed in MATLAB to yield the performance of the equalizer:

```
[number_noeq, ratio_noeq, number_eq, ratio_eq] = multipathadapt(10)
```

The MATLAB script `multipathadapt()` returned the following results. The performance graph is shown in Figure 5.4.

number_noeq = 1055 (# of bits interpreted incorrectly without equalization)
ratio_noeq = 0.5151 (BER without equalization)
number_eq = 38 (# of bits interpreted incorrectly with equalization)
ratio_eq = 0.0186 (BER with equalization)
Elapsed time is 0.00039313 seconds.

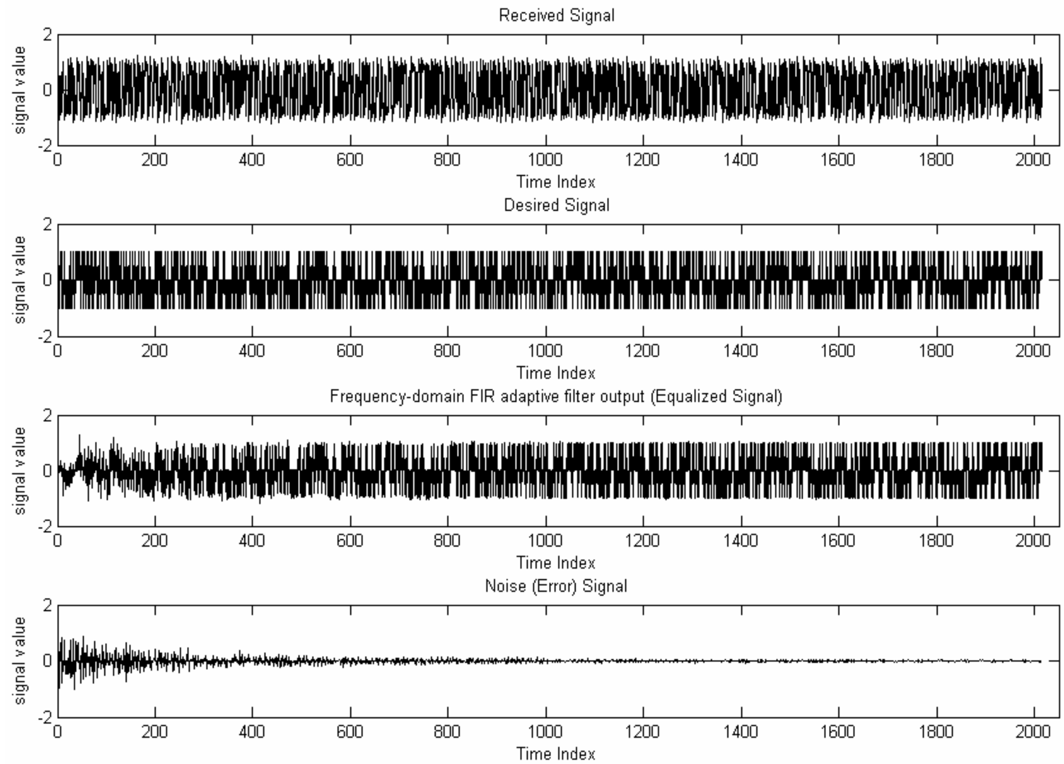


Figure 5.4 Equalizer performance graph for a ten-path Rayleigh fading channel

5.6 Discussions

After running the `multipathadapt()` script for the above four test profiles, BER results were summarized in Table 5.5 below.

Table 5.5 BER comparison table for equalized and non-equalized signal

Number of Paths	Bit Error Rate (BER)	
	Non – equalized	Equalized
3	0.5063	0.0186
5	0.4868	0.0195
7	0.4912	0.0181
10	0.5151	0.0186

Table 5.5 above shows the BER before and after the equalization process for each test profile. BER figures are obtained by the `biterr()` MATLAB function. It can be

observed that the frequency domain adaptive equalizer significantly (approximately 27 times) decreased the BER of the signal in each test profile. The runtime for each test case is given in Table 5.6.

Table 5.6 Runtimes calculated by MATLAB using built-in `tic()` and `toc()` functions

Number of Paths	Runtime (ms)	
	Gateway PC E2300	IBM G40 Laptop
3	0.39543	1.13458
5	0.38990	1.11487
7	0.41397	1.12562
10	0.39313	1.11448

It is vital and necessary to include the specifications of the PC at which the MATLAB script `multipathadapt()` was run on. It was a Gateway E2300 series Desktop PC with a 3.4 GHz Pentium 4 CPU with 1 GB of RAM. Initially, a G40 IBM laptop was used that had a 2.4 GHz CPU with 256 MB of RAM with unacceptable runtimes. GSM is a hybrid system that uses both TDMA and FDMA. In a GSM TDMA frame, each user sends and receives bursts at an interval of 0.577 ms. It was shown that the equalizer can manage to equalize the distorted signal before the next burst of data arrives provided that enough memory and processing power exists. It is also interesting to observe that the ISI is mostly caused by the first three paths arriving immediately after the line of sight signal (Refer to Table 5.5). As the number of paths increase past three, it was observed that the BER of the non-equalized signal does not change significantly.

5.7 Summary

In this chapter, performance graphs as well as the BER comparison table before and after the equalization process of the designed frequency domain adaptive equalizer were presented using four different test profiles. It can easily be observed visually from Figures 5.1 to 5.4 and verified numerically from Tables 5.1 and 5.2 that the implemented frequency domain adaptive equalizer managed to equalize the ISI distorted message signal with acceptable success rates.

CONCLUSION

This thesis had two goals. The first one was to implement a frequency domain adaptive equalizer. The second and the more important one was to investigate the possibility of using this implemented equalizer as an alternative to the traditional MLSE equalizer with Viterbi algorithm used in a GSM receiver. In order to test the performance of the developed adaptive equalizer, a multipath Rayleigh fading channel was modeled using the built-in MATLAB functions. The frequency domain adaptive equalizer was designed to equalize (remove ISI) a GMSK modulated 2072 random bits (14 GSM bursts of 148-bits each) which are passed through a multipath Rayleigh fading channel. The adaptive equalizer is designed using a 32-tap adaptive FIR filter which is trained with a 444-bit preamble training sequence (corresponds to 3 GSM bursts of 148-bits each) using the LMS Gradient algorithm with step size 0.1. On all four different test cases, the equalizer successfully improved the average BER from 0.49985 down to 0.0187 which corresponds to a BER improvement factor of 27. The runtimes of the designed adaptive equalizer varied between 0.38990 to 0.41397 ms. Each TDMA burst is allocated 0.577 ms. Thus, using a powerful enough microprocessor with enough memory (a PC with a similar hardware configuration like the Gateway E2300 Series), it was established that the equalizer can successfully adapt the remaining ISI distorted message signal which is 1628-bit long (corresponds to 11 GSM bursts of 148-bits each). The performance of the equalizer and the LMS Gradient algorithm has been verified in these simulations. The concept of using a frequency domain adaptive equalizer to compensate for ISI created by a multipath Rayleigh fading channel has been proven. In chapter 1, the problem of intersymbol interference and delay spread were explained. In addition, the need for an optimum solution to the problem was stated. In chapter 2, fundamentals of mobile radio propagation were summarized. Rayleigh & Rician multipath fading channels, Doppler frequency, fast and slow fading channels were explained. The correlation between the coherence time and the symbol time was shown. Out of this correlation, the slow and fast fading channel concepts were explained. In chapter 3, equalization basics were introduced as well as different equalization techniques and algorithms.

In chapter 4, the main sections of the adaptive equalizer as well as the multipath Rayleigh fading channel model implemented were explained. Because the FDAE uses the LMS Gradient algorithm for adaptation, it is numerically efficient and requires only 2N + 1 multiplication operations per iteration. In chapter 5, the performance graphs of the implemented equalizer using four different test profiles were presented as well as the BER comparison table before and after the equalization process.

In the future, it would be interesting and valuable to do a real implementation of the frequency domain adaptive equalizer running on a DSP or microprocessor in a GSM receiver and find out the practical improvements necessary. It would also be interesting to test the equalizer with different fading channel models such as Rician, Okumura-Hata, etc. with Doppler shift.

REFERENCES

[1] Theodore S. Rappaport. “Wireless Communications: Principles & Practice”, Second Edition, Prentice Hall, 1999.

[2] Simon Haykin. “Adaptive Filter Theory”, Prentice Hall, Third Edition, 1996.

[3] D. Smalley, “Equalization Concepts: A Tutorial, Application Report”, Atlanta Regional Technology Center, 1994.

[4] D. Falconer and L. Ariyavisitakul. “Frequency Domain Equalization for 2 – 11 GHz Broadband Wireless Systems”, IEEE 80216t-01/01, 2001.

[5] J. J. Shynk. “Frequency-Domain and Multirate Adaptive Filtering”, IEEE 1053-588/92/S3.00, 1992.

[6] S. Johansson. “Adaptive Frequency Domain Equalizer for a Digital Predistortion System”, Master Thesis, Chalmers University of Technology, 2004.

[7] GSM Transceiver Measurements: Laboratory works in Radio Communications, Communications Laboratory, Helsinki University of Technology, 2003.

[8] S. Y. Kung, X. Zhang, and C. L. Myers. “A Recursive QR Approach to Adaptive Equalization of Time-varying MIMO Channels”, Communications and Information Systems, Vol. 5, No. 2, pp. 169 – 196, 2005.

[9] N. Kostov. “Mobile Radio Channels Modeling in MATLAB”, Department of Radio Engineering, Technical University of Varna, Bulgaria, 2003.

[10] R. Landqvist, A. Mohammed. “An Adaptive Block-Based Eigenvector Equalization for Time-Varying Multipath Fading Channels”, Blekinge Institute of Technology, School of Engineering, Sweden, 2004.

[11] H. Cui and P. B. Rapajic. “Complexity Comparison of Iterative Channel Estimation, Equalization and Decoding for GSM Receivers”, The University of New South Wales, Sydney, Australia, 2002.

[12] F. Sainte-Agathe, H. Sari. “New Results in Iterative Frequency-Domain Decision-Feedback Equalization”, France, 2005.

[13] A. Kantsila, M. Lehtokangas, and J. Saarinen. “On Radial Basis Function Network Equalization in the GSM System”, Institute of Digital and Computer Systems, Tampere, Finland, 2003.

[14] K. M. Medapalli. “TDMA Interference Cancellation and Equalization using Tentative Decisions”, Master Thesis, The State University of New Jersey, 1999.

[15] P. Radosavljevic. “Channel Equalization Algorithms for MIMO Downlink and ASIP Architectures”, Master Thesis, Rice University, 2004.

[16] S. Khattak, Dr. Shahid A. Khan, A. Umer, I. Khan. “Space Time Equalization and Interference Cancellation in Frequency Selective MIMO Systems”, COMSATS Institute of Information Technology, Abottabad, 2002.

[17] S. Tjoa. “Adaptive Equalization and the LMS Algorithm”, University of Maryland, 2005.

[18] R. A. Peloso. “Adaptive Equalization: Theory and Practice”, Intellon Corporation, 1999.

[19] M. Garg. “Statistical Wireless Channel Model”, 2005.

[20] B. Sklar. “Rayleigh Fading Channels in Mobile Digital Communication Systems Part I: Characterization”. IEEE Communications Magazine, July 1997.

[21] A. Keyvani. “Modeling and Simulation of a Fading Channel”, Master Thesis, Simon Fraser University, 2003.

[22] J. Rinne. “Digital Communication Through Fading Multipath Channels”, Institute of Technology, Tampere University of Technology, 2005.

[23] G. D. Forney. “The Viterbi algorithm”, Proceedings of the IEEE 61(3): 268–278, March 1973.

[24] R. W., Lucky. “Automatic Equalization for Digital Communications”, Bell System Technical Journal, Vol. 44, pp. 547-588, 1965.

APPENDIX

MATLAB SOURCE CODE

```
function [number_noeq, ratio_noeq, number_eq, ratio_eq] =
multipathadapt(n)
% MULTIPATHADAPT Frequency-domain FIR adaptive filter for time-varying
% multipath fading channel equalization
%
% Ilke Uludag, March 2007

% Number of delay samples
D = 16;

% Sample time (in seconds)
ts = 1/(270.833*10^3);

% Maximum Doppler spread (in Hertz)
fd = 4;

% Path delays (in seconds)
delay = [0 10 20 30 40 50 60 70 80 90 170]*1e-8;
delay = delay(1:n);

% Path gains (in dBs)
gain = [0 -0.9 -1.7 -2.6 -3.5 -4.3 -5.2 -7.8 -10.2 -18 -22.4];
gain = gain(1:n);

% Create Time-varying Rayleigh Multipath fading channel
chan = rayleighchan(ts,fd,delay,gain);
chan.StoreHistory = 1;

% Number of data bits
ntr = 2072;

% Number of training bits
ntr_tr = 444;

% Message Signal
mestx = randint(1,ntr+D);

% Baseband GMSK signal
s = gmskmod(mestx,1);

% Received signal
r  = filter(chan,s);

% Input signal to the filter
x  = r(1+D:ntr+D);

% Desired signal (delayed GMSK signal)
d  = s(1:ntr_tr);

% Noise signal
n = x - d;
```

```
% Initial FFT input powers
del = 1;
% Step size
mu  = 0.1;

% Averaging factor
lam = 0.9;

ha = adaptfilt.fdaf(32,mu,1,del,lam);

tic

[y,e] = filter(ha,x,d);

toc

mesrx_eq = gmskdemod(y,1);
mesrx_noeq = gmskdemod(x,1);
[number_eq,ratio_eq] = symerr(mesrx_eq,mestx(1:ntr));
[number_noeq,ratio_noeq] = symerr(mesrx_noeq,mestx(1:ntr));

subplot(4,1,1); plot(1:ntr-2*D,real(x(2*D:ntr-1))); axis([0 ntr+2 -2
2]);
title('Received Signal');
xlabel('Time Index'); ylabel('signal value');
subplot(4,1,2); plot(1:ntr-2*D,real(d(2*D:ntr-1))); axis([0 ntr+2 -2
2]);
title('Desired Signal');
xlabel('Time Index'); ylabel('signal value');
subplot(4,1,3); plot(1:ntr-2*D,real(y(2*D:ntr-1))); axis([0 ntr+2 -2
2]);
title('Frequency-domain FIR adaptive filter output (Equalized
Signal)');
xlabel('Time Index'); ylabel('signal value');
subplot(4,1,4); plot(1:ntr-2*D,real(e(2*D:ntr-1))); axis([0 ntr+2 -2
2]);
title('Noise (Error) Signal');
xlabel('Time Index'); ylabel('signal value');
```