INTER-CARRIER INTERFERENCE MITIGATION FOR UNDERWATER ACOUSTIC COMMUNICATIONS

BY

ERICA LYNN DALY

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2009

Urbana, Illinois

Adviser:

Professor Andrew C. Singer

ABSTRACT

Communicating at a high data rate through the ocean is challenging. Such communications must be acoustic in order to travel long distances. The underwater acoustic channel has a long delay spread, which makes orthogonal frequency division multiplexing (OFDM) an attractive communication scheme. However, the underwater acoustic channel is highly dynamic, which has the potential to introduce significant inter-carrier interference (ICI). This thesis explores a number of means for mitigating ICI in such communication systems. One method that is explored is directly adapted linear turbo ICI cancellation. This scheme uses linear filters in an iterative structure to cancel the interference. Also explored is on-off keyed (OOK) OFDM, which is a signal designed to avoid ICI.

ACKNOWLEDGMENTS

I am grateful to my adviser, Prof. Andrew C. Singer, for guiding me while conducting this study. My thanks go out as well to my group members, for our enlightening technical discussions. I also thank my parents for their support and upbringing. Finally, I thank my husband Michael for listening to me discuss my research and asking me probing and often fruitful questions about it.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation

High data rate wireless communications are desirable in many underwater applications. For example, searching for underwater mines using a remotely operated vehicle (ROV) is more efficient if the operator has access to streaming video from the ROV. The bandwidth required for streaming video may be obtained by connecting the ROV to the control center via a cable. However, this method is cumbersome, so reliable wireless communication is preferred. Another example is submarine to submarine communication. If submarines wish to communicate without surfacing, they must communicate through the underwater channel; therefore, reliable underwater communications is critical. Improving the data rate at which the submarines communicate allows the submarines to communicate more information for the duration of their communication, which leads to improved situational awareness.

Communicating at a high data rate through the ocean is challenging. Such communications must be acoustic in order to travel long distances. The underwater acoustic channel has a long delay spread, which makes orthogonal frequency division multiplexing (OFDM) an attractive communication scheme. However, the underwater acoustic channel is highly dynamic, which has the potential to introduce significant inter-carrier interference (ICI). This thesis explores a number of means for mitigating ICI in such communication systems.

1.2 Overview

The first five chapters of this thesis present useful background information and motivation for the investigations presented in the last three chapters. Chapter 2 discusses the underwater acoustic channel in detail, which motivates the studies presented in Chapters 6 and 7. Chapter 3 discusses the communication system fundamentals that are the backbone of the systems presented in the rest of this thesis. Chapter 4 discusses turbo equalization. Turbo equalization improves upon the receiver structure presented in Chapter 3 by using feedback from the decoder to boost the performance of the equalizer. Chapter 5 discusses OFDM in detail and how ICI affects it.

The last three chapters discuss novel ICI mitigation techniques for OFDM. Chapter 6 presents a turbo equalization–based ICI cancellation scheme that relies on directly adapted, linear equalizer filters. Chapter 7 explores using information-bearing null carriers to avoid ICI. In particular, it explores folding a symbol with zero power into the signal's constellation. Finally, the thesis is concluded with Chapter 8.

1.3 Notation

1.3.1 Variable naming conventions

Throughout this thesis, the following variable naming conventions are followed:

- Scalar variables are in plain font.
- Vector variables are in bold font.
- Element k of a vector is in plain font with a subscript k. For example, x_k represents element k of vector $\mathbf{x}$
- Matrices are capitalized and in plain font.
- Element j, k of a matrix is capitalized, in plain font, and has a subscript j, k. For example, $A_{j,k}$ denotes the value in the jth row and kth column of matrix A.

- Sequences appear in braces with their range of indices to the right. For example, a sequence of $x[k]$ values where k ranges from 0 to ∞ is denoted by $\{x[k]\}_{k=0}^{\infty}$. The range is omitted for infinite sequences.
- Element k of a sequence is in plain font with a k in brackets. For example, $z[k]$ denotes element k of sequence $\{z[k]\}$.

1.3.2 List of variables used throughout this thesis

Throughout this thesis, the following variables will represent the quantities given here:

$\mathbf{a}$	The vector of information-bearing bits.
$\mathbf{b}$	The vector of encoded bits.
$\mathbf{c}$	The vector of encoded and interleaved bits.
$\mathbf{X}$	The vector of transmitted symbols in the frequency domain. In OFDM, $\mathbf{c}$ is mapped directly to $\mathbf{X}$.
$\mathbf{x}$	The vector of transmitted symbols in the discrete time domain. In standard communication systems, $\mathbf{c}$ is mapped directly to $\mathbf{x}$. In OFDM, $\mathbf{x}$ is the inverse DFT of $\mathbf{X}$.
$\mathbf{h}$	The discrete, baseband, time domain, vector equivalent of the channel.
H	The discrete, baseband, frequency domain, ICI matrix equivalent of the Doppler-corrupted channel.
$\mathbf{y}$	The vector of received values in the discrete time domain.
$\mathbf{Y}$	The vector of received values in the discrete Fourier domain.
$\hat{\mathbf{x}}$	The vector of estimates of the transmitted symbols.
$\bar{\mathbf{x}}$	The vector of estimates of the transmitted symbols that are quantized to the nearest candidate symbol.
$\mathbf{e}$	The vector of symbol estimation errors. $\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}$
e	The mean square error of the estimate $\hat{\mathbf{x}}$. $e = \|\|\mathbf{e}\|\|^2 = \|\|\mathbf{x} - \hat{\mathbf{x}}\|\|^2$
$\hat{\mathbf{c}}$	The vector of estimates of the encoded and interleaved bits.
$\hat{\mathbf{b}}$	The vector of estimates of the encoded bits.
$\hat{\mathbf{a}}$	The vector of estimates of the information-bearing bits.
L	The length of the communicaiton channel.
M	The number of symbols in the transmit signal's constellation.
N	The number of sub-carriers in the OFDM signal.

CHAPTER 2

THE UNDERWATER ACOUSTIC CHANNEL

2.1 Introduction

The underwater acoustic channel provides several unique challenges to the communications engineer. The usable bandwidth of the channel is a decreasing function of both frequency and distance, which severely band-limits long distance unless intermediary relay stations are available [1]. High data rate, long distance underwater acoustic communications communications are wide band signals relative to the carrier frequency, which is typically a few tens of kilohertz [2]. The delay spread is long due to the relatively slow speed of sound underwater. The underwater channel has a significant Doppler spread even when communication is between two stationary entities [3]. It also has a very long propagation delay due to the relatively slow speed of sound, which limits the usefulness of feedback.

The underwater acoustic channel is governed by four main physical parameters: attenuation, surface wave action, sound speed profile, and noise.

2.2 Attenuation

One of the most important aspects of the underwater acoustic channel for communications is its attenuation characteristics. The signal is attenuated via spreading, absorption, and scattering.

Near the transmitter, the signal spreads spherically and therefore decays at a rate of $1/D^2$ due to spreading, where D is the distance from the transmitter. In shallow water with a sandy bottom, the waves encounter two highly reflective boundaries: one with the air and one with the sea

floor. This causes the signal to spread approximately cylindrically after a distance that is much greater than the ocean's depth [4]. This effect can also be observed near the surface of the ocean in deep water because a sound speed gradient exists that traps the sound waves in the upper layer of the ocean. When the spreading is cylindrical, the signal decays due to spreading at a rate of $1/D$.

Unlike terrestrial electromagnetic communications, the propagation medium, seawater, absorbs a significant amount of the transmitted energy. The signal energy decays due to absorption loss as $e^{-\alpha(f)D}$ where $\alpha(f)$ is a function of frequency [5]. As Figure 2.1 illustrates, $\alpha(f)$ is a monotonically

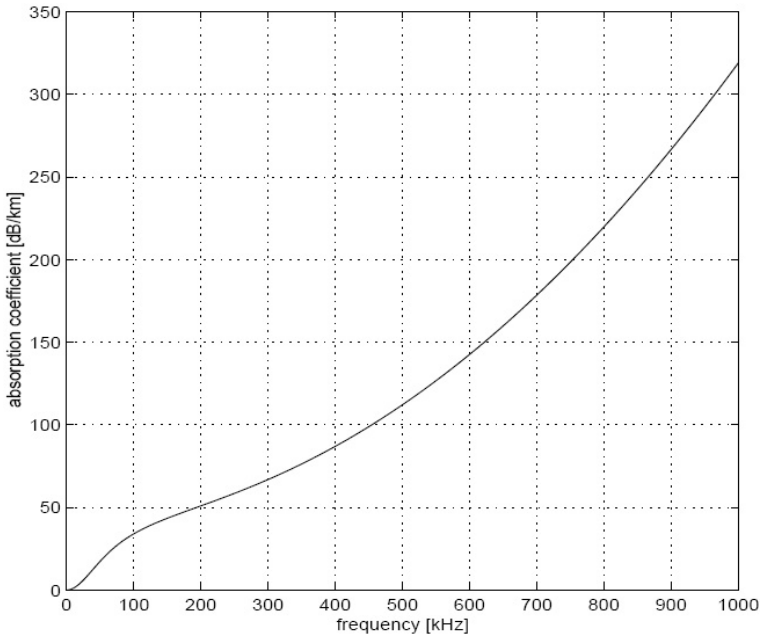


Figure 2.1: The absorption attenuation constant $\alpha(f)$ [1].

increasing function. Due to absorption loss, underwater acoustic communications are effectively band-limited to 100 kHz or less depending on the transmission distance.

Scattering is the third mechanism by which the acoustic signal is attenuated. Scattering is the nonuniform reflection or refraction caused by a nonuniform boundary. Common boundaries in the ocean include bubble clouds, choppy surface waves, and rocky seabeds [6].

2.3 Surface Wave Action

Surface wave action plays a significant role in all but very deep ocean acoustic channels. When the surface wind speed is high, small waves called wind waves form on the surface of the ocean [3]. These waves are superimposed on the larger waves called swell, which are formed over large stretches of ocean by wind energy. The difference between wind waves and swell is that the wind waves are small ripples that respond to local weather conditions, while swell are large waves that are the result of the weather conditions across a large stretch of ocean. Significant surface wave action makes the surface of the ocean choppy, and adds to the scattering of the signal.

Swell are long, large waves on the ocean surface. They are the accumulation of many wind waves over a long distance [3]. They increase in magnitude with increased wind speed and increased unobstructed ocean area upwind of the observation point. Large swell cause a Doppler shift in narrow band signals between even a stationary transmitter and receiver. This is because swell waves are long relative to the wavelength of the transmission, and can be approximated by a moving reflecting plane. The motion of this reflector changes the path length from the transmitter to the reflector to the receiver and applies a time dilation or contraction to the signal. In the frequency domain, this time dilation or contraction manifests itself as a Doppler shift. For a wide-band signal, the signal experiences a frequency shift that is a function of the frequency of the transmitted signal, which is a spreading in frequency.

2.4 Sound Speed Profile

The speed of sound in the ocean is around 1500 m/s. This relatively slow speed of sound decreases the usefulness of feedback. For example, if the source and receiver are 750 m apart, any feedback will be at least one second old. Salient characteristics of the ocean acoustic medium will have significantly changed during that second, so this feedback is quite stale. The slow speed of sound also gives rise to a long delay spread, as a difference in propagation path lengths of one quarter meter results in a

relative delay of 167 μs, the symbol duration of a 6,000 symbols per second signal. The delay spread is often sparse because the signal only encounters two significant surfaces to reflect off of: the ocean surface and floor.

The speed of sound depends strongly on temperature, and less strongly on pressure [3]. The sun's radiation heats the water on the surface. The temperature then decreases until a depth where the sun's rays do not penetrate. At this depth and below, the ocean is a relatively constant, cold temperature. Above this depth is called the thermocline and below is called the deep isothermal layer. In general, the speed of sound underwater increases with increased temperature and pressure [3]. In the thermocline, the pressure increases with depth and the temperature decreases. Because the temperature effect on sound speed is stronger, the speed of sound decreases with depth in this zone [3]. In the deep isothermal layer, the only parameter significantly affecting the sound speed is pressure. Thus sound speed increases with depth as the pressure increases [3]. Very near to the surface, sometimes the speed of sound increases with depth [3]. This is called an inversion layer. This condition occurs when currents circulate the water, making the surface layer a constant temperature. Then, the speed of sound depends only on the pressure, which causes it to increase with depth.

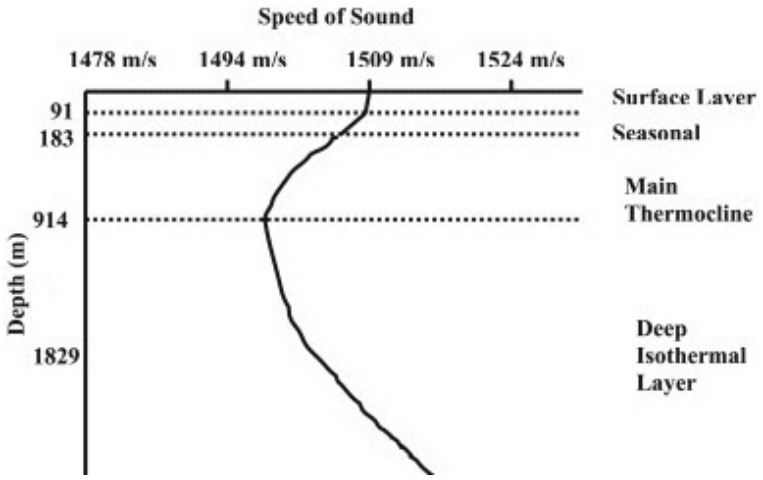


Figure 2.2: The speed of sound as a function of depth [7].

The speed of sound profile, plotted in Figure 2.2, dramatically impacts sound propagation in the ocean. A propagating sound wave refracts, meaning it changes its propagation direction in response to a sound speed

gradient [3]. The wave will bend in the direction of slower sound speed [3]. The result is a continuous bending of the propagation vector in the direction of decreasing sound speed.

Where the thermocline ends and the deep isothermal layer begins, the refraction of sound creates a channel that traps the sound near this boundary [8]. This is because sound waves bend towards the minimum sound speed, which is at the boundary between the thermocline and the deep isothermal layer. This channel is called the deep sound channel or the SOFAR channel. Since the sound propagation in this channel is bounded, it spreads out cylindrically, which leads to less transmission loss than an unbounded channel. Thus, signals can propagate farther in the deep sound channel than at any other depth in the ocean.

When an inversion layer exists, the speed of sound increases with depth for a short distance below the surface [3]. This causes sound waves to bend towards the surface. When the sound wave hits the surface, it reflects [3]. Depending on how steep the gradient of the sound speed is in the inversion layer and the angle with respect to the vertical the reflected sound wave is propagating, the downward propagating reflected wave may refract so much that it begins propagating upwards before it escapes the inversion layer. Thus, the inversion layer acts as a waveguide, and is called a surface duct [8]. Analogously to the deep sound channel, signals propagating in the inversion layer may suffer less transmission loss than unbounded signals.

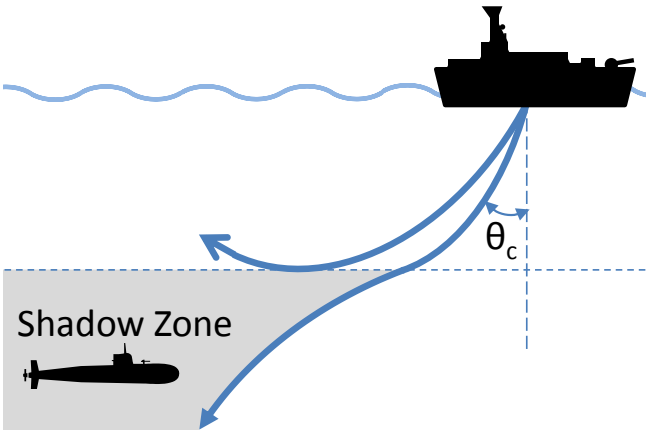


Figure 2.3: Shadow zone involving an inversion layer.

If an inversion layer exists, it is nearly impossible for a transmitter on the surface to send a signal to a receiver in certain regions, called shadow zones

[8]. There is a critical angle where a sound wave sent at this angle will just barely escape being trapped in the inversion layer. This ray will then bend downward as the speed of sound decreases with depth. The shadow zone is bounded by this limiting ray path and the bottom of the inversion layer. Figure 2.3 depicts a shadow zone where communication from a surface ship to a receiver in the shadow zone is nearly impossible. A similar shadow zone exists for a transmitter and receiver located in the thermocline but near the surface. Since the sound rays bend downward, there exists a limiting ray beyond which a receiver is unreachable by a direct path.

Due to sound speed variations with depth, transmitter and receiver placement is crucial for a good communication system. If both are located in the surface duct or near the axis of the deep sound channel, the channel behaves as a waveguide and long range communications are feasible. If the receiver is placed in a shadow zone, the signal will not directly propagate from the transmitter to the receiver, and recovering the message will be challenging.

2.5 Noise

The noise levels underwater are affected by wave action, turbulence, shipping activity, biological activity, and thermal effects. As Figure 2.4 illustrates, the noise underwater is mostly of low frequency. Figure 2.4 shows the noise power spectral density (PSD) for several levels of wave action and shipping activity. The PSD of a wide sense stationary process is the Fourier transform of its autocorrelation. It is a representation of the expected frequency spectrum of the signal. The practical result of the downward-trending noise PSD is that the signal frequency is effectively limited from below. The combination of noise and attenuation pressures leads communication engineers to target center frequencies in the low 10s of kilohertz.

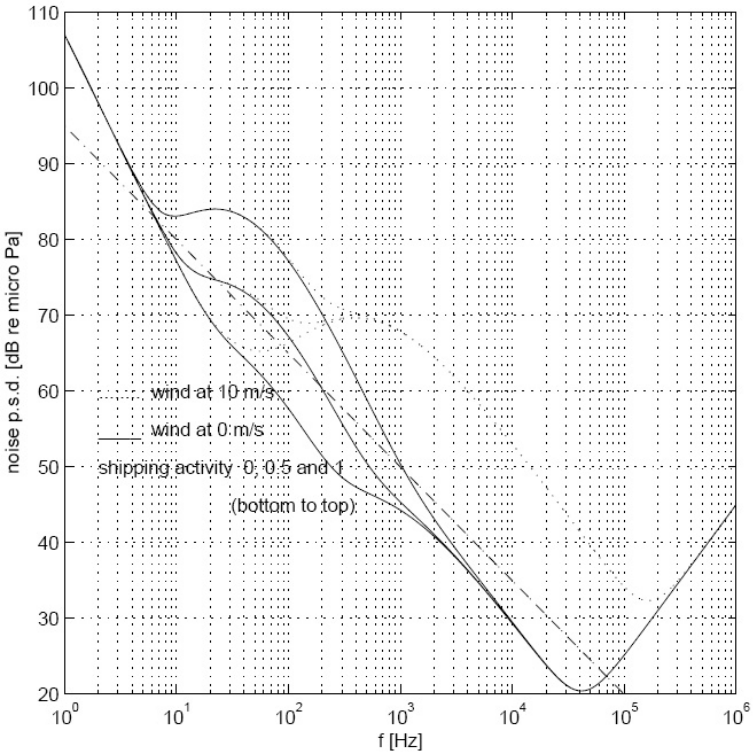


Figure 2.4: A typical noise power spectral density for the ocean acoustic channel [1].

CHAPTER 3

COMMUNICATION SYSTEMS

3.1 Communication System Overview

Communication systems are designed to convey information from the transmitter to the receiver through an imperfect channel. Most practical communication systems we consider in this thesis have a few basic components. The basic communication system is shown in Figure 3.1.

At the transmitter, information bits **a** are first encoded. Encoding adds redundancy by mapping the information bits **a** to a longer bit vector **b**. Encoder design is discussed in detail in Section 3.2. The encoded bits **b** are then interleaved. Interleaving permutes the order of the code bits **b** resulting in bits **c**. The main reason for doing this is to insulate the information bits from bursty noise. In many common codes, especially convolutional codes, the information about bit a_k is contained primarily in a few consecutive bits in **b**. If there were a burst of noise while these bits were transmitted, the information bit a_k could be lost. However, if **b** is interleaved, it is less likely that a noise burst would occur for all the bits in **c** pertaining to a_k. Thus interleaving decreases the probability of error in channels with bursty noise. Interleaving also has an important purpose in turbo equalization, as will be discussed later. The next step in the

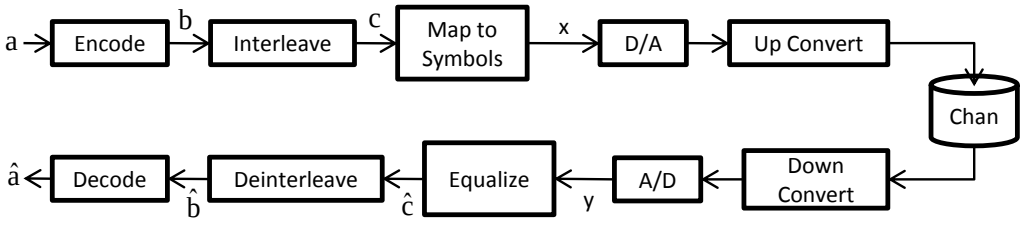


Figure 3.1: Standard communications system diagram.

transmission chain is the symbol mapper. The symbol mapper maps the bits $\mathbf{c}$ into complex symbols $\mathbf{x}$. These digital symbols are then converted into analog symbols. Typically the signal is then up-converted to pass band frequencies by modulating it with a carrier. This is a necessary step for complex symbols. The signal is then ready to be transmitted through the channel.

The channel of interest in this thesis is the underwater acoustic channel. This channel is discussed in detail in Chapter 2. Its salient features are: a significant delay spread, significant variation in time, high latency, low bandwidth, and bursty additive noise [3]. The delay spread gives rise to inter-symbol interference (ISI)—other symbols interfering with the current symbol—even for relatively slow symbol rates in the low kilohertz. The channel is time-varying, which makes it difficult to estimate the channel's transfer function. It also causes a Doppler spread, which is a spreading of the signal in frequency. The high latency of the communication arises from the relatively slow speed of sound, 1500 m/s [3]. Since the latency of the channel is high, the latency of the transmitter and receiver is a less important design goal than in wireless systems. Thus more complex transmitters and receivers are tolerated in underwater communications than in wireless communication systems. The channel is also fairly narrow band with only a few tens of kilohertz of usable spectrum. Finally the noise in the channel tends to be bursty because it is caused in part by biological noises, shipping noises, and bubbles [3].

At the receiver, the operations performed by the transmitter are reversed to recover $\hat{\mathbf{a}}$, an estimate of the information bits. The down-converter mixes the signal back down to baseband. The A/D converter then samples the analog signal, making it digital. Some systems oversample the signal as a hedge against imperfect synchronization between the transmitter and receiver; however, in this thesis all signals are sampled at the symbol rate. At this point, $\mathbf{y}$ is recovered. The signal $\mathbf{y}$ is what would be received if $\mathbf{x}$ were transmitted through the digital baseband equivalent of the channel plus noise. The signal is then equalized. The equalizer attempts to undo the ISI in the received signal to recover the transmitted symbols. It then outputs the bits $\hat{\mathbf{c}}$ associated with those symbols. The vector $\hat{\mathbf{c}}$ may represent hard decisions on the bits or soft decisions. If the equalizer makes soft decisions, it outputs information relating to the probability of the bit

being a 0 or a 1. If the equalizer makes hard decisions on the bits, it quantizes the soft bit decisions and outputs either a 0 or a 1. Equalizers take several forms such as maximum a posteriori probability (MAP) equalizers and linear equalizers. These will be discussed in more detail in Section 3.3. Next, the signal is deinterleaved which is a simple permutation transformation that undoes the transformation the interleaver executed. Finally, the bits are decoded. In this thesis, all bits are encoded by a convolutional code that is then decoded by a MAP decoder employing the BCJR algorithm. This decoder will be discussed in Section 3.4.

3.2 Encoder Design

In general, an encoder is a map between the length N bit vector $\mathbf{a}$ and the length $\frac{N}{R}$ bit vector $\mathbf{b}$. R is the rate of the code, which is always less than one, meaning that $\mathbf{b}$ is always longer than $\mathbf{a}$. The encoded bit vector $\mathbf{b}$ is longer than $\mathbf{a}$ in order to add redundancy to $\mathbf{a}$. Often the code is linear and may be described by

$$\begin{bmatrix} b_1 \\ b_2 \\ \\ \vdots \\ \\ b_{\frac{N}{R}} \end{bmatrix} = \begin{bmatrix} C_{11} & \cdots & C_{1N} \\ C_{21} & & C_{2N} \\ & & \\ \vdots & & \vdots \\ & & \\ C_{\frac{N}{R}1} & \cdots & C_{\frac{N}{R}N} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix}. \tag{3.1}$$

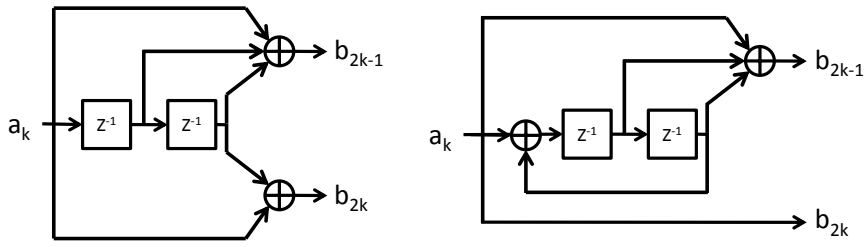
General codes are difficult to decode since the entire block must be decoded at once. Typically structure is added to make decoding easier. The code used in this thesis is a convolutional code. A linear convolutional code may be represented by $\frac{1}{R}$ stacked Toplitz matrices. This is depicted in

Equation (3.2) for a rate $\frac{1}{2}$ code, i.e.,

$$C_{cc} = \begin{bmatrix} c_{11} & 0 & 0 & \cdots & 0 \\ c_{12} & c_{11} & 0 & \cdots & 0 \\ c_{13} & c_{12} & c_{11} & \cdots & 0 \\ & \ddots & \ddots & \ddots & \\ & & & & \\ 0 & & \cdots & 0 & c_{1\kappa} \\ c_{21} & 0 & 0 & \cdots & 0 \\ c_{22} & c_{21} & 0 & \cdots & 0 \\ c_{23} & c_{22} & c_{21} & \cdots & 0 \\ & \ddots & \ddots & \ddots & \\ & & & & \\ 0 & & \cdots & 0 & c_{2\kappa} \end{bmatrix}. \tag{3.2}$$

This matrix describes $\frac{1}{R}$ parallel convolutions of $\mathbf{a}$ with length κ encoding vectors to produce $\mathbf{b}$.

Convolutional codes are designed to produce codewords that are as different as possible. A distance metric between codewords in the vector space in which they reside is used to quantify this difference. Good codewords are found by a nearly exhaustive search [9]. An example of a flow graph representation of a good convolutional code is shown in Figure 3.2(a) [9].



(a) Non-recursive encoder (b) Recursive and systematic encoder

Figure 3.2: Examples of convolutional encoders.

A recursive convolutional code with feedback is depicted in Figure 3.2(b). Feedback from previous register states is used in determining the next state

of the system. Thus, each bit a_k affects all bits in $\mathbf{b}$ after and including bit b_{2k-1}. Often, recursive convolutional codes are also systematic. This means that one of the output coded bit streams is identical to $\mathbf{a}$.

Convolutional codes have a length associated with them. This is denoted by K_{cc}. K_{cc} is equal to the number of delay elements in the flow graph representation of the encoder. For example, the encoders depicted in Figure 3.2 have a length of 2.

3.3 Equalizer Design

The role of the equalizer is to estimate the transmitted symbols $\mathbf{x}$ from the received signal $\mathbf{y}$. For a linear channel, this may be accomplished using a MAP equalizer, which is optimal in terms of minimizing symbol error rate. However, the complexity of this equalizer grows exponentially with channel impulse response length. For this reason, linear equalizers are often used for channels with long impulse responses. Linear equalizers are often simple FIR filters. A third class of common equalizers is that of decision feedback equalizers. These equalizers employ two FIR filters: one that filters the input $\mathbf{y}$ similarly to the linear equalizer and one that filters $\bar{\mathbf{x}}$, the quantized symbol estimate.

3.3.1 MAP equalizer

Ideally, the equalizer would make its decision based on the exact probability that $\mathbf{x}$ was transmitted given $\mathbf{y}$ was received. The vector $\mathbf{x}$ is assumed to consist of iid random variables x_k. Since the information bits a_k are typically iid and their encoded bits are interleaved, this assumption is approximately true. The maximum a posteriori probability (MAP) equalizer makes a decision based on the probability of x_k given the vector $\mathbf{y}$.

For FIR channels and other systems that may be described by a trellis, the likelihoods of the symbols can be computed with an efficient algorithm such as the BCJR algorithm, which is presented later in this chapter [10]. Denote channel's state as s_k. If the channel causes L symbols to interfere with the current symbol, then the channel can be thought of as a register with L slots of memory. The channel's output is then a sum of scaled

versions of the contents of these memory registers and the current input to the channel. The channel state s_k is a vector of the content of the memory registers of the channel. For a linear, time-invariant (LTI) channel, this is simply the previous L transmitted symbols.

Define matrix $A(\chi)$ to be the state transition matrix for valid transitions resulting from $x_k = \chi$. Each column represents the previous state of the channel and each row represents the next state of the channel. Let S represent the number of channel states. S equals M^L, where M is the number of points in the transmit symbol constellation. The state probability matrix A marks valid transitions with a one and invalid transitions with a zero. Also define $P_k = P(x_k = \chi)P(y_k|x_k = \chi)$. This represents all the probability information about the current symbol that the equalizer has available. P_k is a $L \times L$ square matrix. Like $A(\chi)$, each column represents the previous state of the f and each row represents the next state of the channel. There are N matrices A_k, where N is the number of input symbols x_k in $\mathbf{x}$. The first term in P_k represents information about the current symbol generated by the decoder. At the first pass of the turbo equalizer, there will be no input from the decoder, and this value will be $\frac{1}{M}$. The second term represents the probability of receiving y_k given a hypothesis about the transmitted symbols. To calculate this value, symbols $[x_k, ..., x_{k-L}]$ are convolved with the channel transfer function to yield the output of the channel due to the current hypothesis without noise. Then the probability is calculated as the probability of the noise equaling the difference between y_k and the noise free received value due to the hypothesis. The channel transfer function is assumed to be known exactly at the receiver.

The BCJR algorithm is also called the forward-backward algorithm because of the way the likelihoods are calculated. First, the algorithm runs forward through the trellis starting at the beginning. It calculates the probability of x_k given y_j where $j < k$. It then runs backward through the trellis starting from the end, calculating the probability of x_k given all $\mathbf{y}$ following k. These two conditional probabilities of x_k are then merged to form the desired $P(x_k = \chi|y)$ where χ is in the set of employed symbols. The BCJR algorithm works as follows [10, 11]:

1. Initialize f_0 and b_N to length S vectors of ones. Vectors f_k and b_k

represent the probabilities of arriving at each state given the past and future states respectively.

2. Update f_k starting with $k = 1$ and iterating through $k = N$:

   $f_k = P_{k-1} f_{k-1}$.

3. Update b_k starting with $k = N - 1$ and iterating through $k = 0$:

   $b_k = P_k^T b_{k+1}$.

4. Assuming the symbols are BPSK, the likelihood of symbol x_k is:

   $L(x_k|y) = ln(\frac{f_k^T \times (A_k(+1) \cdot P_k) \times b_{k+1}}{f_k^T \times (A_k(-1) \cdot P_k) \times b_{k+1}})$

Note that the calculation of $L(x_k|y)$ involves two types of multiplication. The multiplication denoted by "$\times$" is standard matrix multiplication while the multiplication denoted by "$\cdot$" is element wise multiplication. The division is scalar division.

3.3.2 Linear equalizer

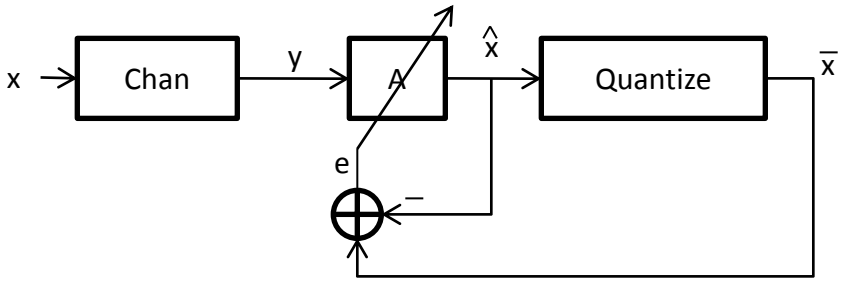


Figure 3.3: Linear equalizer.

A linear equalizer, illustrated in Figure 3.3, attempts to undo the effect of the channel with a filter. This filter may be calculated from a channel estimate or it may be adaptively determined. When there is significant noise in the system, the filter is typically determined to be the one which minimizes the squared error between the transmitted symbols and the filter output.

Let the equalizer filter taps be a vector $\mathbf{A}$. Let the filter output be $\hat{\mathbf{x}}$. Recall that the transmit symbols are $\mathbf{x}$. The minimum mean squared estimate (MMSE) minimizes the following cost function [12]:

$$e = \mathrm{E}\left[\|x - \hat{x}\|^2\right]. \tag{3.3}$$

The function E[$\circ$] denotes expectation with respect to the channel noise. This may be approximated using the LMS update. Since the cost function is quadratic, it is strictly convex with one solution. The LMS algorithm approximates a gradient descent approach to the optimization problem. It updates the equalizer filter as follows [12]:

$$\mathbf{A_{k+1}} = \mathbf{A_k} + 2\mu\tilde{\mathbf{y}}^H e_k. \tag{3.4}$$

The vector $\tilde{\mathbf{y}}$ represents the vector of y_i where i is all integers between $k - L$ and k. The constant L is the length of the equalizer filter. The variable μ is the step size. It may be a constant, or it may vary with time. A larger μ results in a greater update each time step. The value of μ must be carefully chosen to be large enough to allow the filter to be responsive to any variation in the filter taps; yet it must be small enough to converge. e_k is the instantaneous error $\|x_k - \hat{x}_k\|^2$. During training, x_k is available to the receiver. When x_k is not available, the hard symbol decision $\hat{x}_k$ is substituted for x_k. In this way, e_k is approximated by $\|\bar{x}_k - \hat{x}_k\|^2$. This is a good approximation when symbol errors are uncommon.

3.3.3 Decision feedback equalizer

When the signal to noise ratio is high, a decision feedback equalizer (DFE), illustrated in Figure 3.4, performs better than a linear equalizer [12]. A DFE uses the estimates it made of previous symbols, $\bar{\mathbf{x}}$, to cancel their effect on the current symbol. It does this by filtering $\bar{\mathbf{x}}$ through a filter $\mathbf{B}$ and adding this to the filtered received signal.

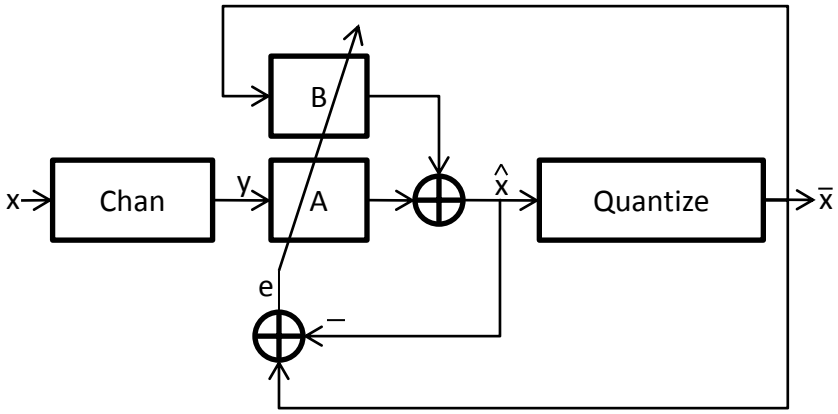


Figure 3.4: Decision feedback equalizer.

3.4 Decoder Design

The function of the decoder is to use the error correction code imposed on the information bits by the encoder to estimate the information-bearing bits. This may be accomplished in a variety of ways and depends on the code used. In this thesis, only a MAP decoder for a convolutional code is explored.

A MAP decoder of convolutionally encoded data is analogous to a MAP equalizer. The MAP equalizer attempts to recover symbols $\mathbf{x}$ that have been filtered and corrupted by noise. The MAP decoder attempts to recover information bits $\mathbf{a}$ from $\hat{\mathbf{b}}$. The bits $\mathbf{b}$ are a filtered version of $\mathbf{a}$ as shown in Figure 3.2. The bits $\hat{\mathbf{b}}$ are the corrupted version of $\mathbf{b}$ that is recovered from the equalizer's estimate of the transmit symbols, $\hat{\mathbf{x}}$.

The data may be decoded efficiently with the BCJR algorithm. The decoder BCJR algorithm is the same as the equalizer BCJR algorithm except that P_k and $\mathbf{A}$ are different, and the initializations f_0 and b_N may be different. The decoder outputs the probability of the bits $\mathbf{b}$ and the estimate of the information bits $\hat{\mathbf{a}}$. The state transition probability matrix $A(\alpha)$ is different when calculating $\mathbf{a}$ and $\mathbf{b}$. When calculating $\hat{\mathbf{a}}$, A is defined to be the state transition matrix for valid transitions resulting from $a_k = \alpha$. When calculating $\hat{\mathbf{b}}$, A is defined to be the state transition matrix for valid transitions resulting from $b_k = \beta$ for every $\frac{k}{R}$th value. For example, decoding a rate $\frac{1}{2}$ convolusitonal code with the BCJR algorithm involves two distinct A matrices—one for the even code bits and one for the odd

code bits. Previously P_k came from two sources: y_k and the probabilities fed back from the decoder. The decoder is only input probabilities from the equalizer. This probability is equal to the probability of a_k given $\hat{\mathbf{b}}$. For the rate $\frac{1}{2}$ code, this equals $\hat{P}(b_{2k-1} = \beta_1)\hat{P}(b_{2k} = \beta_2)$ since the observables b_{2k-1} and b_{2k} indicate that a certain a_k was input to the decoder. P_k is the same regardless of whether $\hat{\mathbf{a}}$ or $\hat{\mathbf{b}}$ is being calculated. Assuming that the convolutional code begins and is terminated to state zero, $f_0 = b_N = [1, 0, 0, ..., 0]^T$. Terminating a convolutional code is desirable because it offers the bits $a[k]_{N_b - K_{cc}}^{N_b}$ at the end of the length N_b bit sequence $a[k]_1^{N_b}$ the same protection from error as the rest of the bits in the bit stream.

CHAPTER 4

TURBO EQUALIZATION

4.1 Turbo Equalization Overview

Turbo equalization is an equalizion and decoding strategy that is designed for channels with inter-symbol interference (ISI) [10]. A turbo equalizer is capable of achieving performance to that which a similar receiver would achieve if the signal were transmitted through an ISI-free channel at the same SNR [10]. It does this by iteratively passing messages about the bit probabilities between the equalizer and decoder. Figure 4.1 shows an overview of a turbo equalizer.

A turbo equalizer may use any type of equalizer and decoder so long as that equalizer and decoder are soft-in, soft-out (SISO) devices. This means that they use bit probabilities (or soft information) to form soft decisions about the bits. For example, the MAP equalizer and decoder described in Chapter 3 are SISO devices.

The BER optimal receiver would perform the equalization and decoding in one massive step. However, due to complexity issues, typical receivers

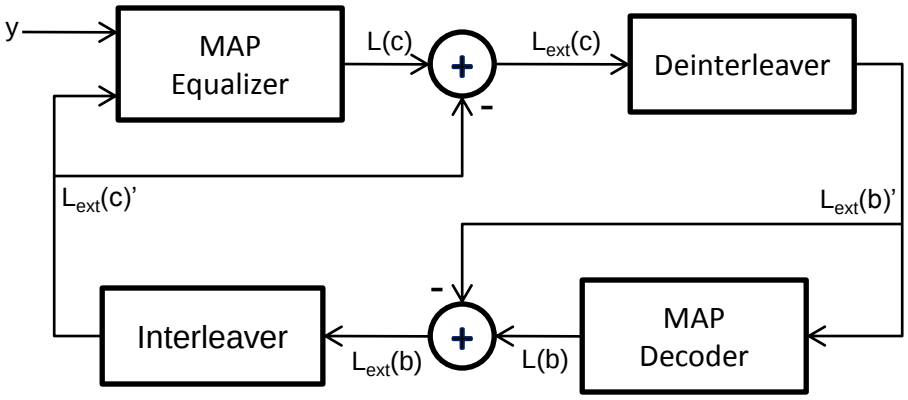


Figure 4.1: Turbo equalizer system diagram.

perform separate equalization and decoding tasks. If the receiver performs the equalization and then the decoding task just once, it disregards information it could have used to make more confident symbol estimates. Primarily, it is ignoring the structure of the error correction code when equalizing. The equalizer pretends that the coded bits that are determining the symbols it is equalizing are independent and equally probable. However, there is unexploited structure in those bits due to the error correction code. Capturing this structure while using a separate equalizer and decoder is possible, and this is the key to turbo equalization. The turbo equalizer first equalizes the symbols normally. It then decodes the symbols normally. After this step, rather than making a hard decision on the bits as a typical receiver would, the decoder feeds soft information about the coded bits back to the equalizer. Due to the structure of the code, the decoder has more information about these bits than the equalizer gave it. Then the equalizer equalizes the signal again using the additional information given to it by the decoder. The bit probabilities the equalizer outputs the second time should be more accurate than those it output the first time since it now has access to information gleaned from the structure in the error correcting code. Next, the decoder decodes using the updated soft bits from the equalizer. It then outputs soft information that may be used by the equalizer. The process continues in this way until a stopping condition is reached. At this time, the soft decisions output by the decoder are quantized to hard bit decisions, which are then output from the receiver. This last step is not shown in Figure 4.1 for clarity.

The soft bit decisions may not be passed directly from the equalizer to the decoder or from the decoder to the equalizer even if the bits are not interleaved [10]. The soft decision $L(c_k)$ output from the equalizer must not depend on the soft information about bit c_k that the equalizer previously passed to the decoder. Similarly for the decoder, $L(b_k)$ must not depend on the soft information about bit b_k the decoder gave to the equalizer the previous iteration. The soft information that was not influenced by previously output soft information from the same device is referred to as "extrinsic information." The principle of only exchanging extrinsic information is referred to as the "turbo principle."

The soft decision about a given bit that is used for turbo equalization is typically a log likelihood ratio (LLR). The LLR is simply the logarithm of

the ratio of the likelihoods of the hypotheses given the observation. For example, the LLR of the information bits a given the received signal y is given by

$$L(a_n) = ln\frac{P(a_n = 1|y)}{P(a_n = 0|y)}. \tag{4.1}$$

The first hypothesis is that $a_n = 1$ and the second is that $a_n = 0$. Using LLRs rather than bit probabilities makes some of the processing easier. If the probability of a_n being 1 is greater than the probability of it being 0, then $L(a_n) > 0$. Otherwise, $L(a_n) \leq 0$. Also, for the special case of a MAP decoder, the extrinsic LLR can be computed from

$$L_e xt(b_n) = L(b_n) - Lext(b_n)'. \tag{4.2}$$

Similarly for a MAP equalizer, Equation (4.2) describes the extrinsic LLR, replacing b_n with c_n.

The extrinsic information as defined above has a flaw. Ideally, the extrinsic information passed between the equalizer and decoder would contain no information that the equalizer or decoder respectively passed to its peer during all previous iterations. As defined above, the extrinsic information is only devoid of information output during the preceding iteration. However, input information affects the calculation of multiple soft decisions. If there were no interleaver, information fed into the decoder about bit b_k would be used to calculate the soft outputs of neighboring bits including b_{k+1}. Since the symbol x_k corresponding to bit b_k interferes with symbol x_{k+1}, information about b_k (which is the same as info about x_k for BPSK) affects the calculation of the output soft estimate of b_k. Thus, even though only extrinsic information is used, the turbo principle is violated when there is no interleaver. The obvious solution to this problem is to use an interleaver. The interleaver ensures that these short cycles of information do not exist. With a well designed interleaver, although b_k and b_{k+1} are correlated, the information about the bits arriving at the equalizer $L(c_k)$ and $L(c_{k+1})$ are probably not correlated. With an infinite block of data and a random interleaver, $L(c_k)$ and $L(c_{k+1})$ are most likely uncorrelated since the probability of interleaving two neighboring bits to the same neighborhood is infinitesimal. With a finite length data block, cycles longer than one iteration may occur. When this happens, the turbo principle is

violated and the turbo equalizer ceases to yield gains. The block length N required for a given channel and convolutional code combination to not violate the turbo principle for Ψ turbo iterations is roughly bounded by

$$N \geq ((L-1)(K_{cc}-1))^{\Psi}. \tag{4.3}$$

At every equalization step, the information about bit k influences the calculation of $L-1$ bits other than bit k, where L is the length of the channel. At every decoding step, information about bit j influences the calculation of $K_{cc}-1$ bits in addition to bit j. Thus, at every turbo iteration, $(L-1)(K_{cc}-1)$ bits are affected by information about bit k in addition to the bit k, where K_{cc} is the constraint length of the convolutional code. If the number of bits N is finite, after enough iterations, information about bit k must influence its calculation through other bits. Equation (4.3) relates the maximum number of iterations following the turbo principle to N for the best possible interleaver.

4.2 EXIT Charts

Extrinsic information transfer (EXIT) charts are a valuable tool for predicting the performance of the turbo equalizer for a given channel and error correcting code [13]. EXIT charts plot the mutual information between the extrinsic LLR output of the equalizer and the correct bits versus the mutual information between the input to the equalizer and the correct bits. The same is plotted for the decoder, except that the axes are swapped. The mutual information input to the decoder is plotted versus the mutual information output from the decoder. This transposition is useful because the extrinsic output of the encoder becomes the input to the decoder and vice versa. Denote the mutual information output from the equalizer as I_{eq} and the mutual information output from the decoder as I_{dec}.

If the turbo principle is followed, i.e., the bit stream is infinitely long and is randomly interleaved, then the EXIT chart predicts the performance of the turbo equalizer [10]. In this case, the output of the equalizer the initial time the symbols are equalized will have mutual information given by $I_{eq}(0)$

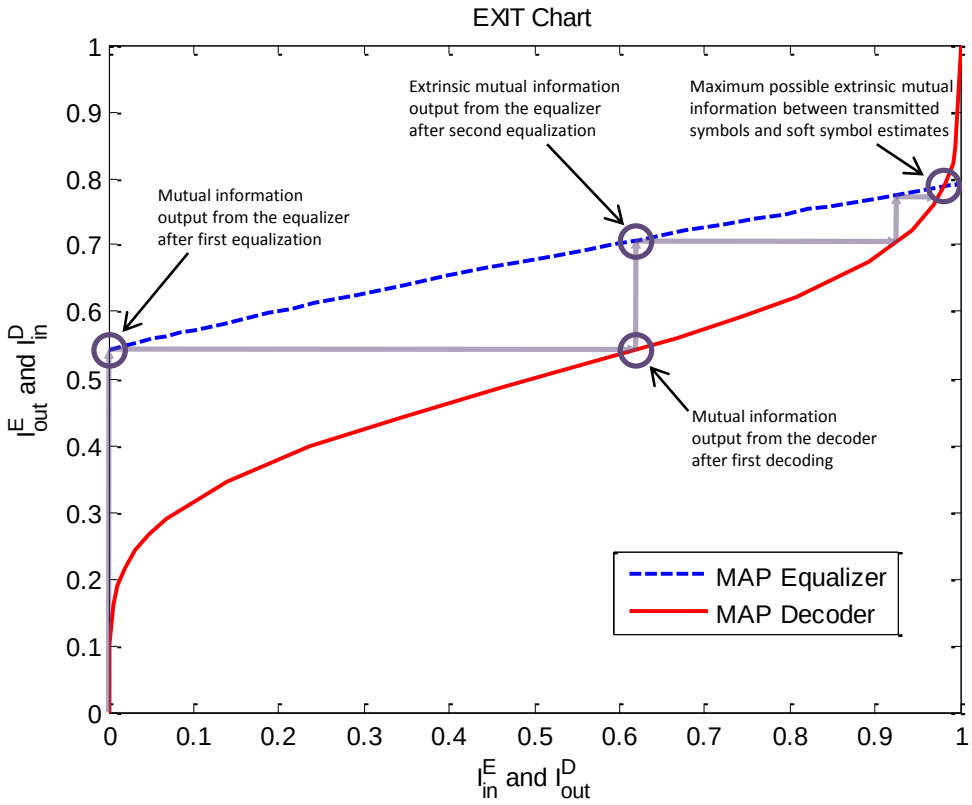


Figure 4.2: EXIT chart for a turbo equalizer. The SNR is 4 dB. The channel is an ISI channel with h=[0.407 0.815 0.407]. The code used is a length 3 convolutional code with generator G=[7 5].

since no information was input to the equalizer initially. The input to the decoder is I_{eq}, so the output of the decoder will be $I_{dec}(I_{eq}(0))$. Then this extrinsic information is input to the equalizer. The equalizer's output then has extrinsic information $I_{eq}(I_{dec}(I_{eq}(0)))$. The turbo equalizer thus traces a zig zag path through the exit chart. The path alternates between the equalizer's vertical path and the decoder's horizontal path. These paths terminate at the intersection of the lines I_{eq} and I_{dec}.

An example of an EXIT chart is shown in Figure 4.2. The impulse response of the channel is $h = [0.407, 0.815, 0.407]$, which yields $y_k = .407x_{k-1} + .815x_k + .407x_{k+1} + w_k$, where w_k is the noise at time k. The SNR is 4 dB. A rate $\frac{1}{2}$ convolutional code is used as an error correcting channel code. The code has constraint length 3 and generator [7 5]. This code is depicted in Figure 3.2(a) on page 14.

CHAPTER 5

ORTHOGONAL FREQUENCY DIVISION MULTIPLEXING (OFDM)

5.1 Overview of OFDM

OFDM is a type of frequency division multiplexing (FDM). FDM routs a transmitted bit stream into several parallel bit streams and transmits each by modulating them onto carriers of different frequencies. Because the entire OFDM signal may then be modulated again to bring it from baseband to passband, these carriers are termed subcarriers. In order to avoid inter-carrier interference (ICI), which is interference from another subcarrier, each modulated subcarrier spectrum must have zero energy at all subcarrier frequencies but its own. This may be accomplished with several signal pulse shapes. The signal pulse may be rectangular. Then the spectrum of each individual modulated subcarrier in this case is a sinc with nulls at multiples of $\frac{1}{\tau}$ Hz, where τ is the time duration of the pulse. Therefore, subcarriers may be spaced at multiples of $\frac{1}{\tau}$ Hz without inherent ICI. A raised cosine pulse may similarly be used. These pulses have lower sidelobes in the frequency domain, which is advantageous if the precise subcarrier frequency is not known at the receiver. Typically, rectangular pulses are used because the modulation may then be efficiently accomplished via a fast Fourier transform (FFT). The advantage of using FDM over single carrier modulation is that the same amount of information can be transmitted through N virtual parallel channels at a symbol rate that is $\frac{1}{N}$ the rate of a single carrier system, where N is the number of used subcarriers. Each of these virtual channels is a single tap channel if the physical channel is LTI.

OFDM transforms a LTI channel with inter-symbol interference (ISI) into a set of parallel, frequency non-selective channels in a bandwidth–efficient way [14]. The time domain noise-free output of the channel is composed of

a convolution of the transmitted signal and the DFT of the impulse response of the channel. If this convolution were instead a cyclic convolution, in the discrete Fourier domain, the noise-free output of the channel is composed of an element wise multiplication of the transmitted signal and the impulse response of the channel. Thus, the ISI channel is turned into one-tap parallel channels. The convolution is transformed into a cyclic convolution by repeating the last several time domain symbols of each N–symbol OFDM block at the beginning of the block. The repeated portion of the signal is called the cyclic prefix. The cyclic prefix must be as long as or longer than the impulse response of the channel. The parallel channel gains are the DFT of the time domain channel response.

OFDM transmission is implemented as follows: Bits are routed from serial to N parallel channels. Each channel is independently modulated. For future reference, call this signal vector $\mathbf{X}$. These N symbols are transformed to the time domain via an IFFT. The cyclic prefix is added to the signal by copying the last L samples and appending them to the beginning of the signal. L must be larger than the delay spread of the channel. This signal may then be up-sampled and modulated. It is then transmitted and received. At the receiver, any modulation or up-sampling is undone. Then the cyclic prefix is discarded. The signal is then transformed into the frequency domain via an FFT. Call this signal vector $\mathbf{Y}$. The equivalent channel from $\mathbf{X}$ to $\mathbf{Y}$ is then N parallel channels, each with its own gain and additive noise. The gain in each channel may be estimated using a variety of techniques. Known training data may be transmitted, which the receiver can use to estimate the channel. This training data could occupy one of several OFDM symbols in a block. Then the channel gain is estimated by simply dividing $\mathbf{Y}$ element-wise by $\mathbf{X}$. The training data could also take the form of pilots, which are reserved subcarriers in each OFDM symbol that carry training data. Assuming that there are much fewer significant time domain channel taps than there are pilot tones, the frequency domain channel transfer function may be accurately estimated from these few pilot tones. This is because the channel frequency response has fewer degrees of freedom than there are independent samples of the response. Once the channel frequency response is estimated, the symbols may be recovered by dividing the channel gain associated with each frequency from the received signal at each frequency.

Another benefit of OFDM is that it captures frequency diversity, which is important in a fading environment [14]. In such an environment, carriers at certain frequencies are severely attenuated, making accurate communication impossible. However, not all carriers in an OFDM system will experience a deep fade at the same time. With coding across the subcarriers, the information lost in the fades is potentially recoverable.

5.2 System Model of OFDM

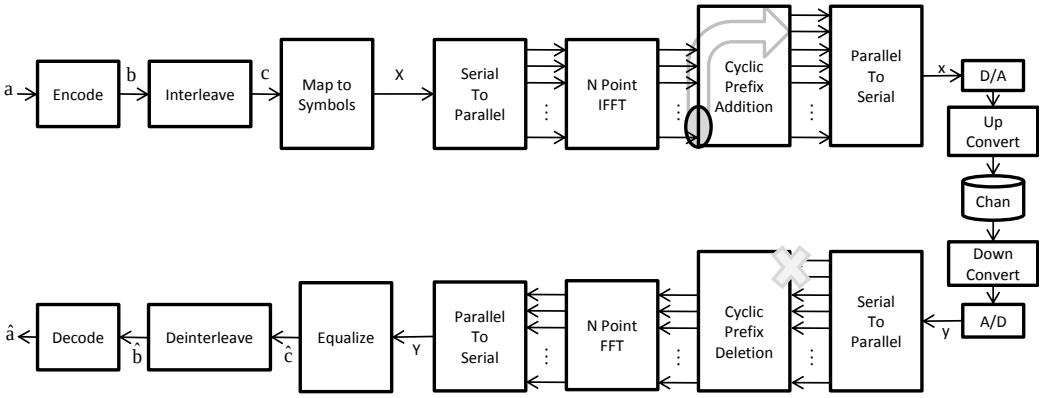


Figure 5.1: OFDM communication system.

In this section, the basic OFDM system model is described. The OFDM system model is shown in Figure 5.1. The information bits **a** are encoded by a recursive, systematic convolutional encoder, producing code bits **b**. These bits are interleaved forming c and then mapped onto M-ary symbols **X**. Pilot symbols may then be injected into the symbol stream. These symbols are known at the receiver and are added in order to learn the channel. The symbol stream is converted into N streams which are processed by the inverse discrete Fourier transform. A cyclic prefix is then added to the signal to prevent ISI. Adding a cyclic prefix involves copying the last several samples of the signal and appending them to the beginning of the signal. The cyclic prefix should be longer than the delay spread of the channel to prevent ISI and to simulate a cyclic convolution of the signal with the channel. This signal is then up-converted, transmitted through the channel and down-converted by the receiver.

At the receiver, the cyclic prefix is discarded, and the remaining signal is

processed using the discrete Fourier transform (DFT). If the channel were linear and time-invariant, the time domain signal corresponding to each OFDM symbol now appears to have been cyclically convolved with the channel thanks to the cyclic prefix. Once the signal passes through the DFT, the cyclic convolution is transformed into a simple multiplication of the frequency domain channel transfer function with the symbols $\mathbf{X}$. Thus, the ISI channel is transformed into N independent channels described by

$$Y_l(n) = H_l(n)X_l(n) + W_l(n), \tag{5.1}$$

where $\mathbf{X}$ is the vector of transmitted symbols. The vectors $\mathbf{Y}$ and $\mathbf{W}$ are the received symbols and noise respectively at the output of the DFT. The vector $\mathbf{H}$ is the equivalent baseband frequency domain channel. The index l ranges from 0 to $N-1$ and represents discrete frequency. The index n is a discrete time index that is sampled at the OFDM symbol rate. The OFDM symbol rate is equal to N times the symbol rate, since N symbols are transmitted per OFDM symbol.

When the channel is time-varying, the received signal in the lth frequency bin may be modeled by

$$Y_l(n) = \sum_{k=l-L}^{l+L} H_{k,l}(n)X_k(n) + W_l(n), \tag{5.2}$$

where $H_{k,l}(n)$ is the transfer coefficient between the symbol in channel k and the symbol in channel l, and L is the number of channels to either side of channel l that interfere with channel l. Due to the properties of the DFT, X_k and $H_{k,l}$ are periodic functions of k with a period of N, meaning, for example, that $H_{-1,l}(n)X_{-1}(n) = H_{N,l}(n)X_N(n)$. The quantity $W_l(n)$ is the lth component of the discrete Fourier transform of the received noise.

CHAPTER 6

DIRECTLY ADAPTED LINEAR TURBO EQUALIZED OFDM

6.1 Directly Adapted Linear Turbo Inter-Carrier Interference Cancellation Technique

6.1.1 Receiver structure

The proposed ICI canceling system is illustrated in Figure 6.1. The OFDM signal is processed normally until it arrives at the equalizer: first, the cyclic prefix is removed, and then the signal encounters an FFT. After the FFT, the signal is processed by a bank of directly adapted linear filters, producing the symbol estimate, $\hat{\mathbf{X}}$. The likelihoods of the bits conveyed by $\mathbf{X}$ are calculated. These are then deinterleaved and decoded by a MAP decoder. The output soft bits are again interleaved and are used to generate $\bar{\mathbf{X}}$. At this point, either a new turbo cycle begins or the cycle is halted by a stopping criterion. At the conclusion of the process, information bits are derived from the bit probabilities calculated by the decoder.

6.1.2 Directly adapted ICI cancellation

The function of the filter bank in Figure 6.1 is to undo the interference caused by neighboring sub-channels in the signal. Accordingly, frequency indices $i - L$ through $i + L$ are used to calculate $\hat{X}_i$.

First, the signal is processed in N streams corresponding to the N sub-channels within the OFDM symbol. Each stream is linearly filtered across frequency through the feed-forward filter f_i. Simultaneously, $\bar{X}_i$ is filtered across frequency by the feedback filter b_i. The soft symbol, $\bar{X}_i$ is the expected value of symbol i given the bit probabilities output from the decoder. In order to refrain from violating the turbo principle, the soft

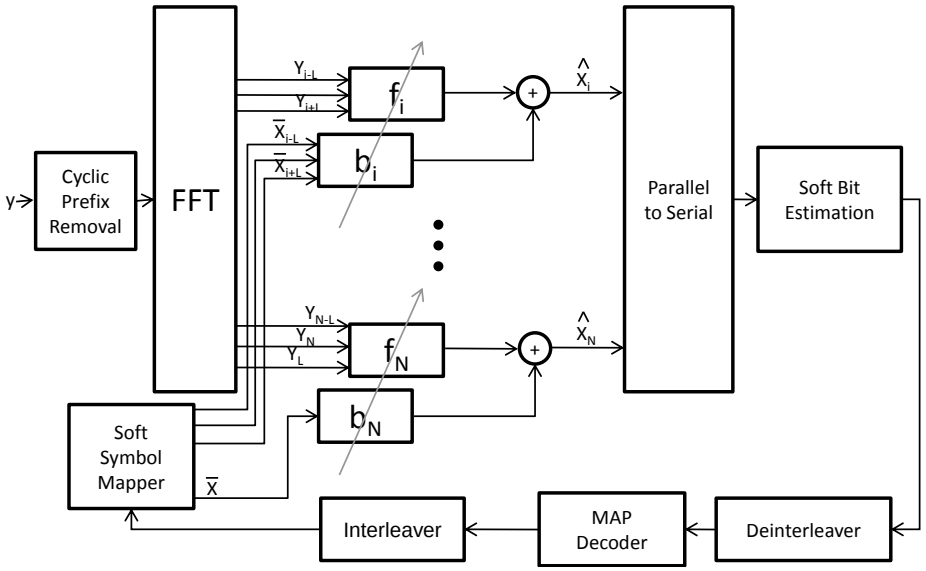


Figure 6.1: Directly adapted turbo ICI canceler block diagram.

symbol $\bar{X}_i$ is disregarded when processing $\hat{X}_i$. The sum of the two filter outputs constitutes the symbol estimate, $\hat{X}_i$.

The LMS algorithm is used to update the filter coefficients. The LMS update is standard, given by Equation (6.1) for f_i and by Equation (6.2) for b_i:

$$f_i(n+1) = f_i(n) + \mu Y_i(n) e_i^*(n), \tag{6.1}$$

$$b_i(n+1) = b_i(n) + \mu \bar{X}_i(n) e_i^*(n). \tag{6.2}$$

Here, e_i is defined as error between X_i and the estimate $\hat{X}_i$. Each of the f_i and each of the b_i filters is estimated independently. These $2N$ LMS algorithms approach the MMSE optimal filter taps in time, where the time index n is incremented every OFDM symbol period. It is assumed that the ICI transfer function imposed by the channel varies slowly in time so that the channel estimator can track it.

The symbol likelihoods are calculated using e_i, which is assumed to be a zero mean Gaussian random variable with variance σ_e^2. The likelihood that symbol S was transmitted, given that X_i was received, is assumed to be of the form:

$$P(X_i = S|\hat{X}_i) = \frac{1}{\pi\sigma^2} e^{\left(-\frac{|S-\hat{X}_i|^2}{2\sigma^2}\right)}, \tag{6.3}$$

where σ^2 is the variance of the true error $X_i - \hat{X}_i$. In high SNR, Equation

(6.3) may be approximated by

$$P(X_i = S|\hat{X}_i) = \frac{1}{\pi\sigma_e^2} e^{-\frac{|e_i|^2}{\sigma_e^2}}. \tag{6.4}$$

The proposed algorithm uses Equation (6.4) to calculate the symbol likelihoods since the transmitted symbol S is not known by the receiver. The symbol likelihoods are converted into bit likelihoods by summing the likelihoods of the symbols in which the bit of interest is in one of the two possible states. This information is then fed to the decoder.

6.2 Simulations

6.2.1 Setup

A synthetic channel is used to model the effects of ICI in the underwater acoustic channel in a series of simulations. Approximately one million bits are used in each simulation. A rate 1/2 recursive convolutional code with generator polynomial $(5,7)_8$ is used for forward error correction. The bits are mapped to a 256 channel OFDM symbol employing 8-PSK. The cyclic prefix is assumed to be long enough to completely avoid ISI. No pilots are used.

The channel is generated directly as an ICI transfer function, which is a linear map between transmitted signals i and received signals k. The relationship between X and Y is given by

$$Y = \begin{bmatrix} 0.815 & 0.407 & 0 & \cdots & 0.407 \\ 0.407 & 0.815 & 0.407 & \cdots & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ 0.407 & 0 & \cdots & 0.407 & 0.815 \end{bmatrix} X + N. \tag{6.5}$$

The matrix representing the ICI transfer function has a banded structure with all its energy contained within L places of the main diagonal, where L is 1 in this case. Although the underlying time domain channel is varying with time, the ICI transfer function remains static for the duration of the simulation. This is a reasonable model for the underwater acoustic channel.

Surface waves are the primary cause of Doppler spread when the transmitter and receiver are not moving. These waves have periods on the order of several seconds. Thus, on the time scale of each packet, the surface of the water is moving approximately at a constant velocity. This would induce a static ICI transfer function because the derivative of the communication path lengths is constant.

A directly adapted turbo ICI canceler is simulated on this channel. The algorithm employs L-tap filters, where L corresponds exactly to the parameter L of the channel. The algorithm performs 10 turbo iterations before completing. The number of iterations were chosen based on the length of the bit stream. Recall the interleaver discussion from Section 4.1. One million bits were simulated. For a three-tap channel and a convolutional code of constraint length 3, Equation (4.3) yields $(2 * 2)^{10} \approx 1,000,000$. This means that it is possible for a bit stream of one million bits to not violate the turbo principle for 10 iterations.

Also, a standard OFDM receiver is simulated on this channel for comparison. It assumes that the channels are orthogonal, meaning that the ICI transfer matrix is diagonal. It estimates the channel ideally as the diagonal of the true ICI matrix. Since the channels are assumed orthogonal, the symbol estimates are calculated from the received frequency domain signal by dividing this signal by the channel estimate. Analogously to the turbo ICI canceling receiver, soft bit information is gleaned from the symbol estimates. These are then fed into a MAP decoder, which makes a hard decision on the transmitted bits.

6.2.2 Results

Figure 6.2 shows the result of the simulation of the turbo equalizer. Although the performance starts out poor for all SNR, with increasing turbo iterations, it converges to the ICI-free lower bound for E_b/N_0 greater than 5 dB.

Figure 6.3 shows the result of the simulation on an EXIT chart for an E_b/N_0 of 7 dB. The convergence of the algorithm, which is shown in Figure 6.2, corresponds nicely to the convergence predicted by the EXIT chart. This confirms that the simulated block length of a million bits was long

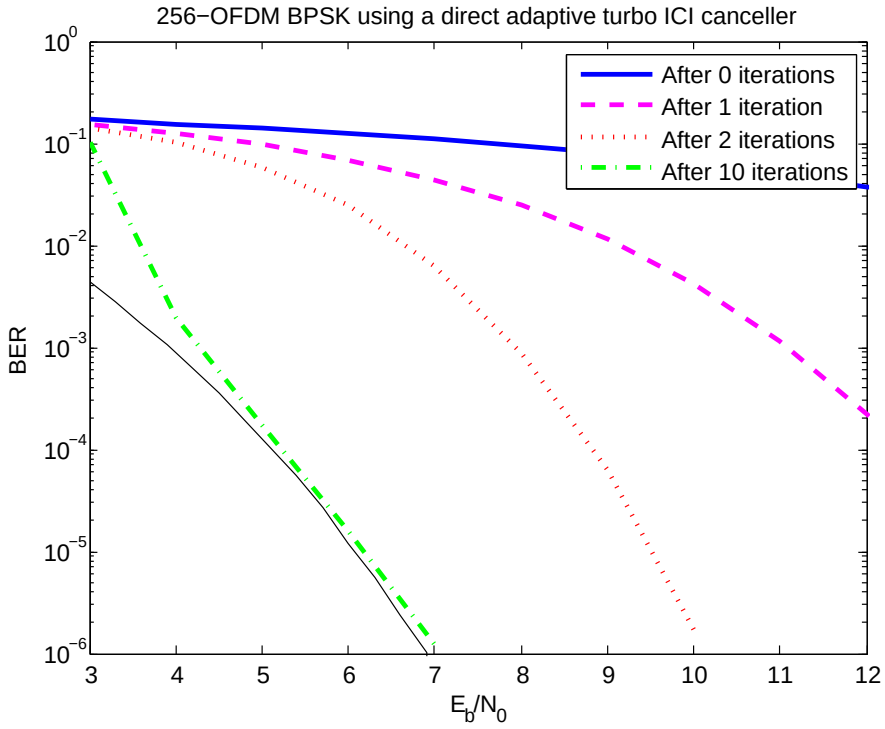


Figure 6.2: Simulation results for turbo equalized 256-OFDM using BPSK.

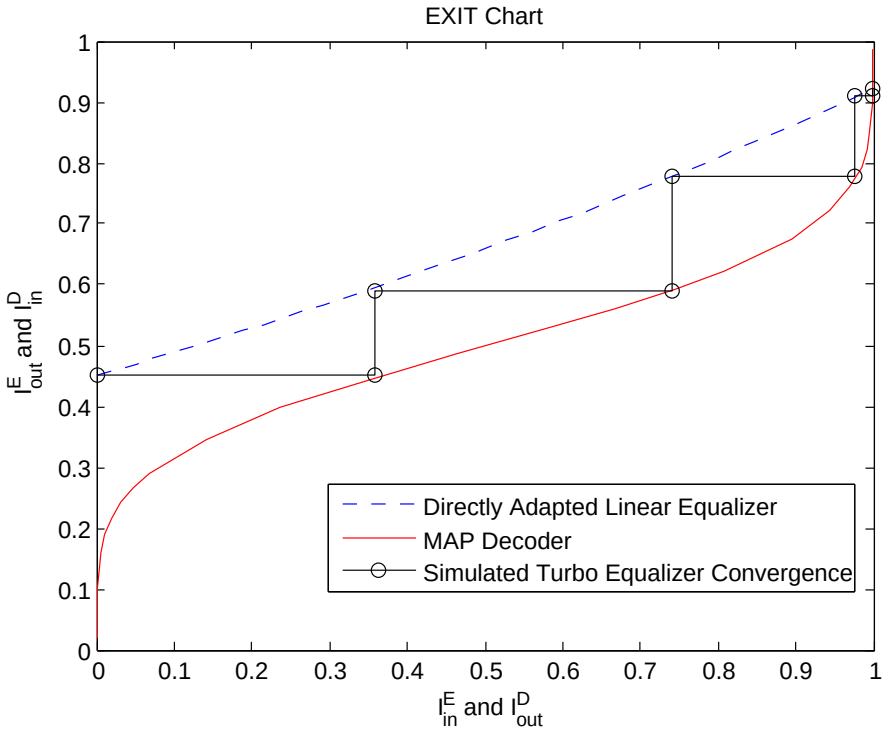


Figure 6.3: EXIT chart for directly adapted linear, turbo equalized 256-OFDM using BPSK. The channel is described by Equation (6.5) and has 7 dB E_b/N_0.

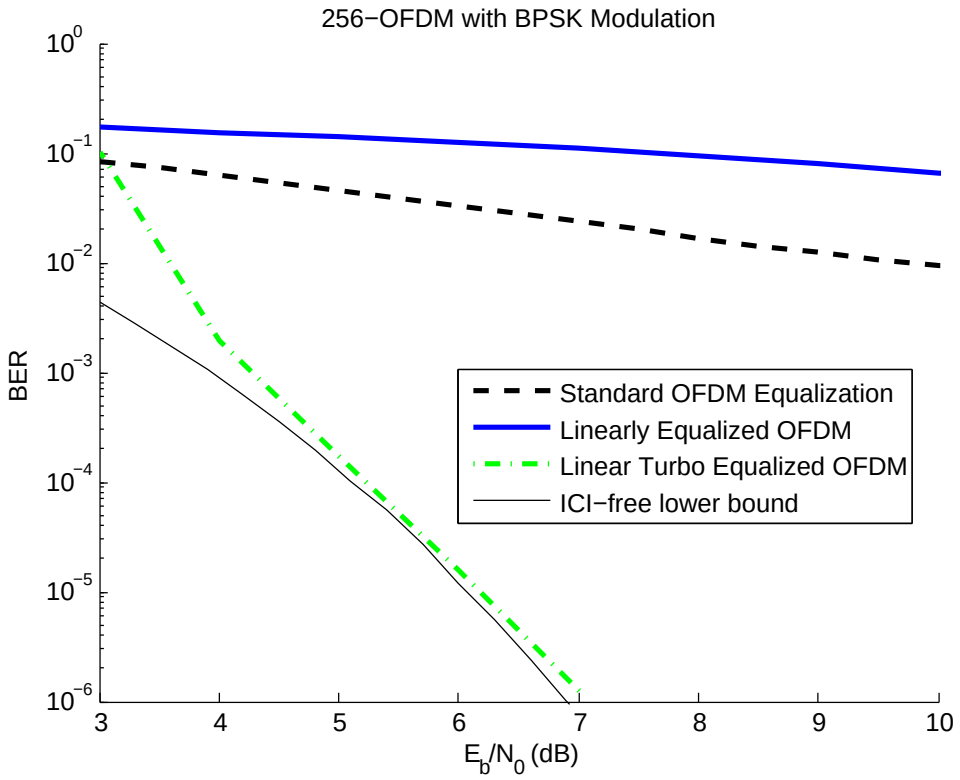


Figure 6.4: Simulation results for 256-OFDM using BPSK.

enough to capture the full benefit of the turbo equalizer for an E_b/N_0 as low as 7 dB.

Figure 6.4 compares the turbo equalizer's performance to the performance of other OFDM equalizers. Standard equalized OFDM had consistently poor performance. Since the ICI is severe in this simulation and the standard OFDM receiver does not address it, the ICI is the dominant cause of bit errors, which causes a BER floor of about 0.1. Also shown in Figure 6.4 is the linear ICI cancelling equalizer, which is equivalent to the turbo equalization algorithm after 0 turbo iterations. Finally, the turbo equalized OFDM signal is shown after 10 turbo iterations. This scheme vastly outperforms the two standard schemes.

CHAPTER 7

ON-OFF KEYED (OOK) OFDM

7.1 OOK OFDM

When dealing with inter-carrier interference (ICI) due to Doppler spread, there are three angles of attack: jointly estimating the symbols in the presence of ICI, removing the effect of ICI before applying normal receiver techniques, and avoiding ICI in the first place. On-off-keyed (OOK) OFDM attempts to avoid ICI. It communicates information by transmitting either a signal or silence on each subcarrier. This has the effect of placing nulls in random locations and thereby avoiding ICI originating in the sub-channels containing the nulls. Unlike typical null-padded OFDM, OOK OFDM does not necessarily lower the bit rate. The signal that is transmitted when the signal is "on" may be a pure sinusoid that conveys no information itself, a PSK signal, or any type of modulated signal. In this thesis, the focus is on using PSK to convey additional information during the "on" time intervals.

It is desirable to communicate the same amount of information each symbol, so that bit errors are local rather than possibly affecting the entire bit stream following the error. This is not automatic with OOK signals that are modulated since an "on" symbol conveys more information than an "off" symbol. Thus, rather than encoding information in the "on"/"off" state independently of the modulation of the "on" state, the information is jointly encoded by augmenting the modulating signal's constellation with a null constellation point. A null is a constellation point at the origin of the constellation space. If the modulation has $2^M - 1$ points in addition to the null, the composite signal has a power of two number of states and an integer number of bits is encoded into each symbol. If the number of constellation points in the composite signal is not a power of two, blocks of bits must encode several symbols to achieve a non-integer number of bits

encoding each symbol.

7.2 Simulation Setup

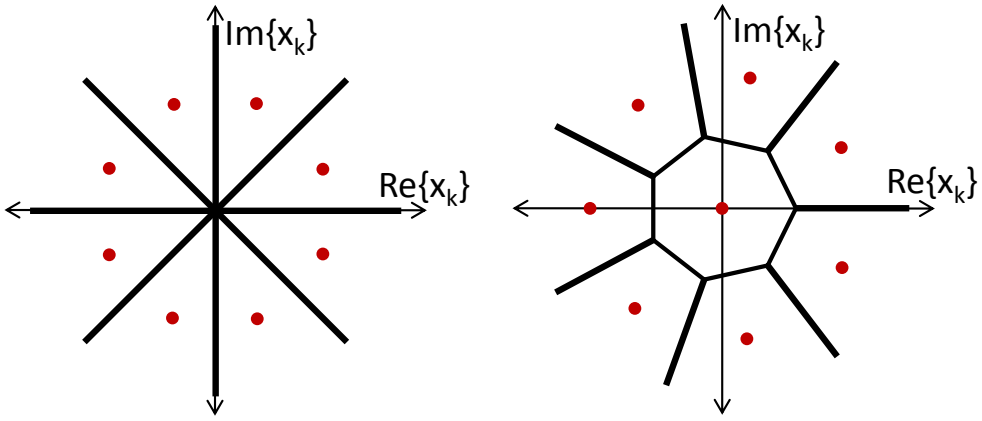
OOK OFDM was simulated using additive white Gaussian noise (AWGN) and a ICI matrix that does not vary with time. The frequency domain input to the channel X is related to the frequency domain output of the channel Y as

$$Y = \begin{bmatrix} 0.20 & 0.96 & 0 & \cdots & 0.20 \\ 0.20 & 0.96 & 0.20 & \cdots & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ 0.20 & 0 & \cdots & 0.20 & 0.96 \end{bmatrix} X + N. \tag{7.1}$$

The following modulations were investigated: standard 8-PSK and augmented 7-PSK. Recall that the augmented M-PSK signal refers to a standard M-PSK constellation augmented by a null. The constellation diagrams of 8-PSK and augmented 7-PSK are shown in Figure 7.1 for comparison. Also depicted in Figure 7.1 are the decision boudaries for the nearest neighbor symbol estimator. Augmented 7-PSK is equivalent to an OOK signal with a $\frac{1}{8}$ chance of being "off" that transmits a 7-PSK signal when it is "on." The signals were received by a standard OFDM equalizer with perfect channel knowledge. This equalizer divides the received frequency domain signal Y element-wise by the diagonal entries of the ICI matrix and treats the interference as noise. A rate $\frac{1}{2}$ recursive convolutional code with generator polynomial $(5,7)_8$ is used for forward error correction.

7.3 Results

Augmented 7-PSK is an excellent choice for OOK OFDM, because adding a null to 8-PSK increases the minimum distance between constellation points. Indeed, the comparison of bit error rates shown in Figure 7.2 shows that augmented 7-PSK outperforms standard 8-PSK modulated OFDM with ICI in the high SNR regime. For the chosen channel, the augmented 7-PSK signal does not exhibit a BER floor to the precision of the simulation, while



(a) 8-PSK. (b) Augmented 7-PSK.

Figure 7.1: Constellation diagrams of the simulated signals.

the 8-PSK signal clearly does.

Figure 7.3 compares the performance of 8-PSK and augmented 7-PSK for an ICI-free channel. As shown in Figure 7.3, augmented 7-PSK marginally outperforms 8-PSK without ICI in the high SNR regime and underperforms 8-PSK in the low SNR regime. In the high SNR regime, the probability of symbol error is well approximated by its upper bound given by

$$P_e < (M-1)Q\left(\sqrt{\frac{d_{min}}{2N_0}}\right), \tag{7.2}$$

which is a function of the minimum distance between codewords [15]. Due to the geometry of the signal constellation, for a given average symbol energy, the augmented 7-PSK signal has a higher minimum Euclidean distance d_{min} between constellation points than does 8-PSK. Thus, it has a lower bit probability in the high SNR regime. In the low SNR regime, the probability of symbol error depends on the full geometry of the problem, and the probability of symbol error is greater for the augmented 7-PSK than for the 8-PSK.

The BER floor behavior of the signal in ICI can be explained using Figure 7.3. In the high SNR regime, the signal to interference plus noise ratio (SINR) is approximately equal to the signal to interference ratio. The interference scales with the power of the signal, so the signal to interference ratio is constant. When interference is treated as AWGN noise, there is a

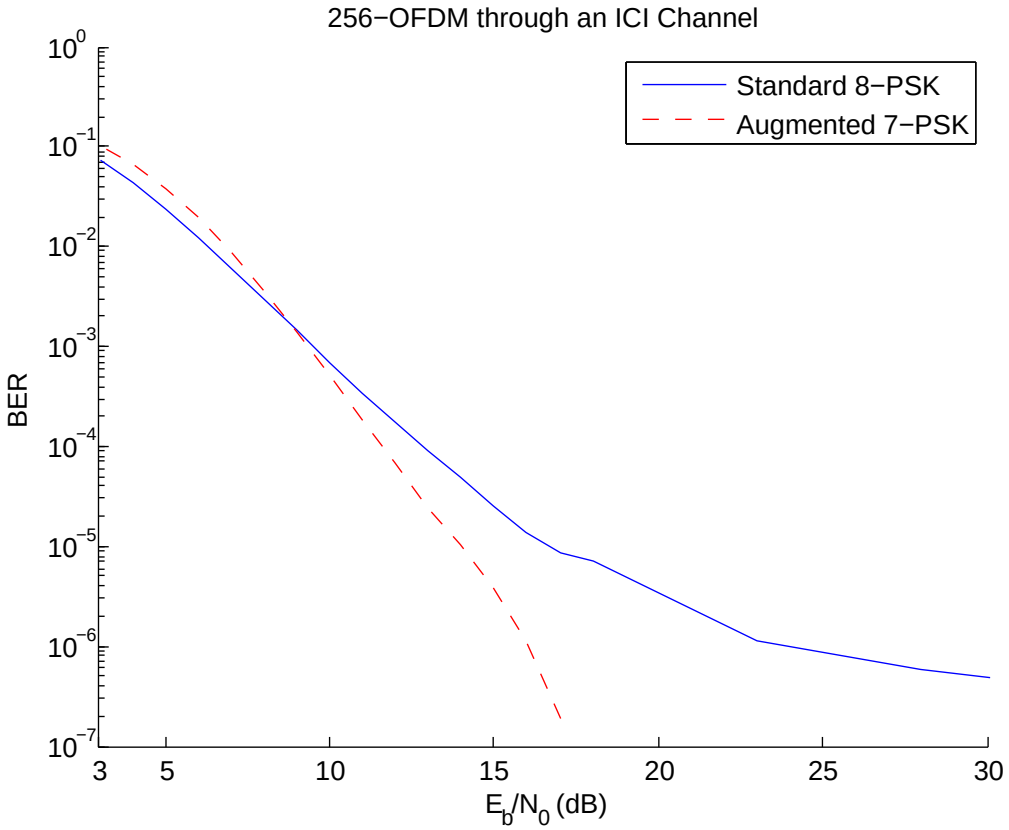


Figure 7.2: Results of simulated 256-OFDM signals using 8-PSK with ICI.

BER floor at the BER level of the ICI-free case in SNR equal to the signal to interference ratio of the ICI-corrupted signal. For the same channel, the augmented 7-PSK signal has a larger signal to interference ratio than 8-PSK does, since the nulls do not contribute interference to the signal. This is in addtion to augmented 7-PSK outperforming 8-PSK even at the same level of signal to interference ratio, yielding a much lower BER floor for augmented 7-PSK in ICI.

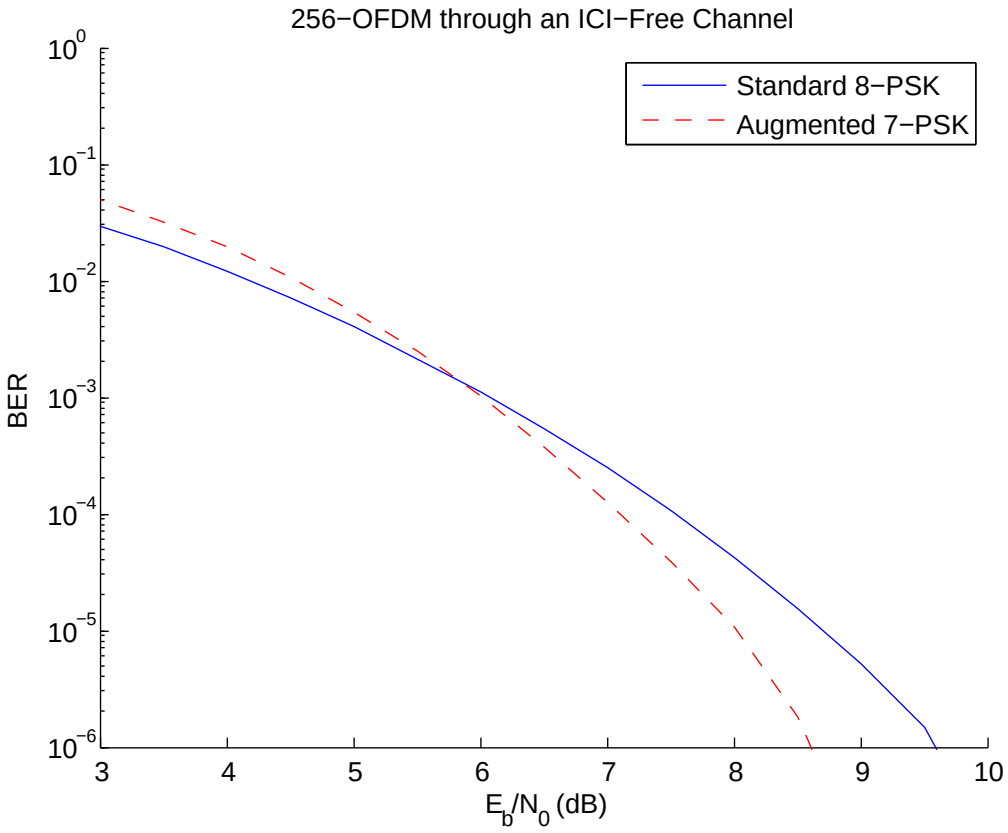


Figure 7.3: Results of simulated 256-OFDM signals using 8-PSK without ICI.

CHAPTER 8

CONCLUSIONS

8.1 Conclusions of Research

Two techniques for mitigating ICI in the underwater acoustic channel were shown. The first, directly adapted linear turbo ICI cancellation, canceled the ICI at the receiver. It is a turbo equalizer that equalizes across frequency instead of time. The second, OOK OFDM, diminished the amount of ICI the channel caused by inserting nulls into the signal. Since these nulls conveyed information, their use did not necessarily lower the bit rate.

ICI was shown to be successfully canceled using a directly adapted turbo OFDM ICI canceler. This ICI canceler was demonstrated in a high ICI environment, and it achieved the ICI-free lower bound on performance for an E_b/N_0 greater than 5 dB. This ICI canceler is ideal for mitigating ICI caused by the underwater acoustic channel, since it does not rely on the transmitter having knowledge of the channel, and it also does not rely on a separate channel estimation algorithm.

OOK OFDM outperforms standard OFDM in certain cases. OOK OFDM was demonstrated using 7-PSK that was augmented with a null. This signal is equivalent to an OOK signal with a $\frac{1}{8}$ chance of being "off" that transmits a 7-PSK signal when it is "on." A standard OFDM equalizer with perfect channel knowledge was used to equalize the signal. It was shown that, in an ICI-free channel, augmented 7-PSK outperforms standard 8-PSK for an E_b/N_0 greater then 6 dB. Most importantly, in a channel with ICI, augmented 7-PSK has a much lower BER floor than standard 8-PSK. Since the transmission is unreliable below 6 dB of E_b/N_0, augmented 7-PSK outperforms standard 8-PSK in situations where it makes sense to use the computationally efficient, standard OFDM equalizer.

8.2 Discussion of Future Work

Future investigations include applying turbo equalization to OOK OFDM and exploring OOK OFDM signals with arbitrary null probabilities. Also, turbo equalization on short blocks of bits will be investigated. This is an important avenue of study for the underwater acoustic channel in particular because practical bit rates through the channel are limited to a few tens of thousands of bits per second. Transmitting a million bits per block as was done in this thesis may be impractical for real time applications since it would take around a minute for the block to pass through the underwater channel.

REFERENCES

[1] M. Stojanovic, "On the relationship between capacity and distance in an underwater acoustic communication channel," *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, vol. 1, no. 4, pp. 34–43, October 2007.

[2] M. Stojanovic, "Recent advances in high-speed underwater acoustic communications," *IEEE Journal of Oceanic Engineering*, vol. 21, pp. 125–136, April 1996.

[3] L. Berkhovskikh and Y. Lysanov, *Fundamentals of Ocean Acoustics*. New York, NY: Springer, 1982.

[4] F. Jensen, M. Kuperman, W.A. Porter, and H. Schmidt, *Computational Ocean Acoustics*. New York, NY: Springer-Verlag, 2000.

[5] H. Medwin and C. Clay, *Acoustical Oceanography: Principles and Applications*. New York, NY: Wiley, 1977.

[6] L. Kinsler, A. Frey, A. Coppens, and J. Sanders, *Fundamentals of Acoustics*. New York, NY: Wiley, 2000.

[7] M. Domingo, "Overview of channel models for underwater wireless communication networks," *Physical Communication*, vol. 1, no. 3, pp. 163–182, 2008.

[8] A. Quazi and W. Konrad, "Underwater acoustic communications," *IEEE Communications Magazine*, pp. 24–29, 1982.

[9] P. Frenger, P. Orten, and T. Ottosson, "Convolutional codes with optimum distance spectrum," *IEEE Communications Letters*, vol. 3, no. 11, pp. 317–319, 1999.

[10] R. Koetter, A. Singer, and M. Tchler, "Turbo equalization," *IEEE Signal Processing Mag*, vol. 21, pp. 67–80, 2004.

[11] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, no. 2, pp. 284–287, 1974.

[12] S. Haykin, *Adaptive Filter Theory*, 3rd ed. Upper Saddle River, NJ: Prentice–Hall, 1996.

[13] M. Tuchler, R. Koetter, and A. Singer, "Turbo equalization: Principles and new results," *IEEE Transactions on Communications*, vol. 50, no. 5, pp. 754–767, May 2002.

[14] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. New York, NY: Cambridge University Press, 2005.

[15] J. Proakis, *Digital Communications*, 4th ed. New York, NY: McGraw-Hill, 2001.